1979

A microprocessor controlled digital filter

Robert Wade Walstrom
Iowa State University

INFORMATION TO USERS

This was produced from a copy of a document sent to us for microfilming. While the most advanced technological means to photograph and reproduce this document have been used, the quality is heavily dependent upon the quality of the material submitted.

The following explanation of techniques is provided to help you understand markings or notations which may appear on this reproduction.

1. The sign or "target" for pages apparently lacking from the document photographed is "Missing Page(s)". If it was possible to obtain the missing page(s) or section, they are spliced into the film along with adjacent pages. This may have necessitated cutting through an image and duplicating adjacent pages to assure you of complete continuity.

2. When an image on the film is obliterated with a round black mark it is an indication that the film inspector noticed either blurred copy because of movement during exposure, or duplicate copy. Unless we meant to delete copyrighted materials that should not have been filmed, you will find a good image of the page in the adjacent frame.

3. When a map, drawing or chart, etc., is part of the material being photographed the photographer has followed a definite method in "sectioning" the material. It is customary to begin filming at the upper left hand corner of a large sheet and to continue from left to right in equal sections with small overlaps. If necessary, sectioning is continued again—beginning below the first row and continuing on until complete.

4. For any illustrations that cannot be reproduced satisfactorily by xerography, photographic prints can be purchased at additional cost and tipped into your xerographic copy. Requests can be made to our Dissertations Customer Services Department.

5. Some pages in any document may have indistinct print. In all cases we have filmed the best available copy.

PLEASE NOTE:

In all cases this material has been filmed in the best possible way from the available copy. Problems encountered with this document have been identified here with a check mark ✓.

1. Glossy photographs _____
2. Colored illustrations _____
3. Photographs with dark background _____
4. Illustrations are poor copy _____
5. Print shows through as there is text on both sides of page _____
6. Indistinct, broken or small print on several pages _____ throughout _____
7. Tightly bound copy with print lost in spine _____
8. Computer printout pages with indistinct print ✓
9. Page(s) _____ lacking when material received, and not available from school or author _____
10. Page(s) _____ seem to be missing in numbering only as text follows _____
11. Poor carbon copy _____
12. Not original copy, several pages with blurred type _____
13. Appendix pages are poor copy _____
14. Original copy with light type _____
15. Curling and wrinkled pages _____
16. Other _____

A microprocessor controlled digital filter

by

Robert Wade Walstrom

A Dissertation Submitted to the

Graduate Faculty in Partial Fulfillment of

The Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Major: Electrical Engineering

Approved:

Signature was redacted for privacy.

In Charge of Major Work

Signature was redacted for privacy.

For the Major Department

Signature was redacted for privacy.

For the Graduate College

Iowa State University
Ames, Iowa

1979

TABLE OF CONTENTS

INTRODUCTION

Early applications of digital signal processing were mainly to simulate or approximate the performance of an analog system on a digital computer. In this respect, digital signal processing offered flexibility in modifying the simulated system characteristics. However, the complexity of the digital signal processing calculations required so much computer processing time that the real-time use of digital signal processing technique was impractical.

The advent of the fast Fourier transform and the development of specialized hardware devices, such as hardware multipliers and correlators, drastically reduced the processing time required for digital signal processing functions. This resulting increase in processing speed has made it possible to incorporate digital signal processing functions with real-time applications.

The development of the microprocessor has had a dramatic effect in many areas. Its small size has greatly reduced the complexity of previously computer assisted or controlled systems. Its small cost has not only made it possible to allow computer applications to previously "stand-alone" systems, but has created a whole new class of systems or products which were not feasible before. The microprocessor has become appealing for so many applications because it is a general purpose, low cost device which, by applying unique software, can be adapted to perform a specific function. If the specific function involved is a complex one, the use of a microprocessor will usually result in a simpler hardware configuration than if the function were implemented strictly in hardware.

While the general purpose feature of the microprocessor is advantageous for many, many applications, this is not the case for some. Some applications may require very complex operations to be performed within a relatively short time period. The complex operation must then be broken down into many simple steps, each of which must be broken down into a series of general purpose microprocessor machine language instructions. The time required by the microprocessor to perform this complete series of programming steps may or may not be longer than the time allowed by the requirements of the system being implemented.

Much work has already been done and continues to be accomplished in advancing digital signal processing technology. While the adaptive digital filter system described in this dissertation is a unique approach, its contribution to digital signal processing technology is negligible. The purpose of this dissertation is to examine the application of a general purpose microprocessor to a digital signal processing problem, analyze the performance of the microprocessor in this application, and determine whether a microprocessor is actually appropriate for this type of application.

BACKGROUND AND LITERATURE REVIEW

Digital Filtering

A digital filter is one which transforms an input sequence of numbers into an output sequence of numbers according to a prescribed algorithm. The algorithm is, then, the heart of the digital filter and there are a great number of approaches which may be taken to arrive at an appropriate algorithm. Some of the basic approaches and some resulting realizations will now be discussed.

One approach is to initially design a continuous analog filter according to a given specification. Once the analog filter is designed, it is converted or transformed into digital form usually by performing an s-domain to z-domain transformation on the analog filter transfer function. The digital filter is then implemented with a combination of delayed inputs and outputs, and feed-forward and feedback coefficients. In general, a transfer function is

$$H(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_\ell z^{-\ell}}{1 + b_1 z^{-1} + b_2 z^{-2} + \ldots + b_k z^{-k}}$$

which would correspond to the recursive relationship

$$y_n = a_0 x_n + a_1 x_{n-1} + \ldots + a_\ell x_{n-\ell} - (b_1 y_{n-1} + b_2 y_{n-2} + \ldots + b_k y_{n-k})$$

where a_ℓ denotes a feed-forward factor;

b_k denotes a feedback factor;

x_n denotes an input value; and

y_n denotes an output value.

The z-domain transfer function can be manipulated in many ways to obtain different digital filter configurations. For example, a direct nonrecursive (no feedback elements) filter with an input-output relationship

$$y_n = \sum_{m=0}^{\ell} a_m x_{n-m}$$

is represented in Figure 1. Such a filter is called a finite impulse response (FIR) filter, that is, a filter with an impulse response, h(nT), that is zero outside a finite range, $n < 0$ and $n > N-1$ where N is the number of delayed inputs (2).

A filter which incorporates feedback in its implementation is shown in Figure 2. This filter is a direct recursive realization and has an input - output relationship of

$$y_n = \sum_{m=0}^{\ell} a_m x_{n-m} - \sum_{m=1}^{k} b_m y_{n-m}.$$

Because of the feedback elements, the response of such a filter is generally no longer finite. Such a filter is termed an infinite impulse response (IIR) filter or a filter which has an impulse response, h(nT), of infinite duration (2).

The direct canonic form of recursive filter is shown in Figure 3. Other canonic forms, such as the parallel canonic in Figure 4, and the cascade canonic in Figure 5, are possible combinations of these forms.

Another general design approach to digital filtering is to directly approximate the desired filter frequency response characteristics.

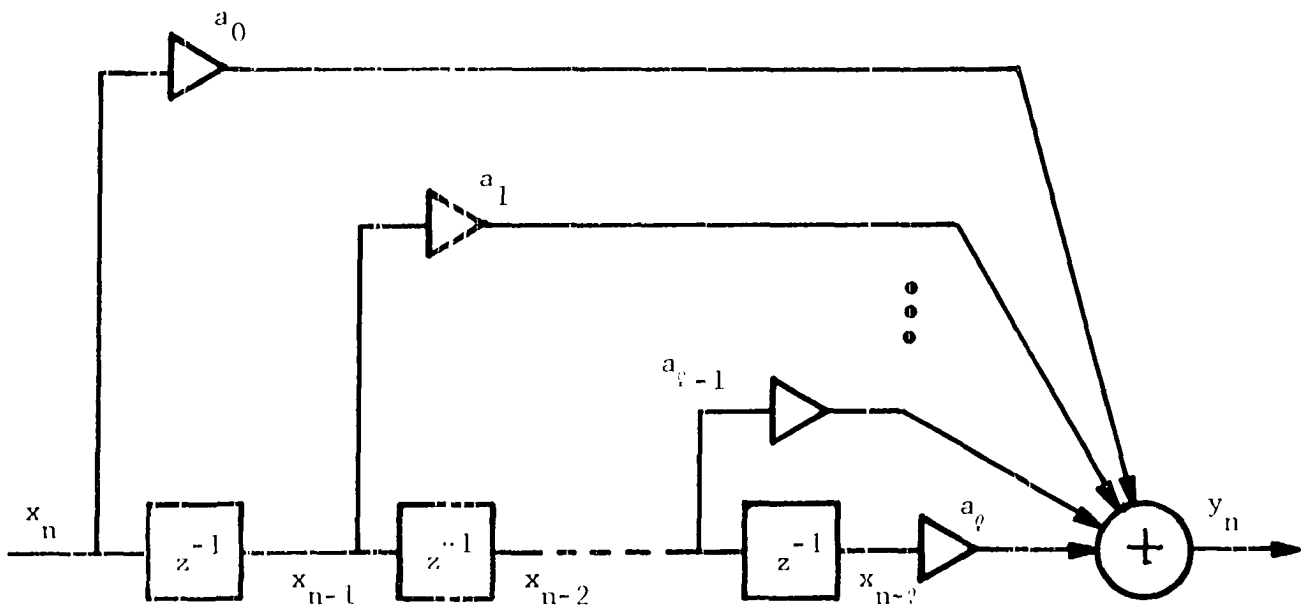


Figure 1. Direct nonrecursive digital filter

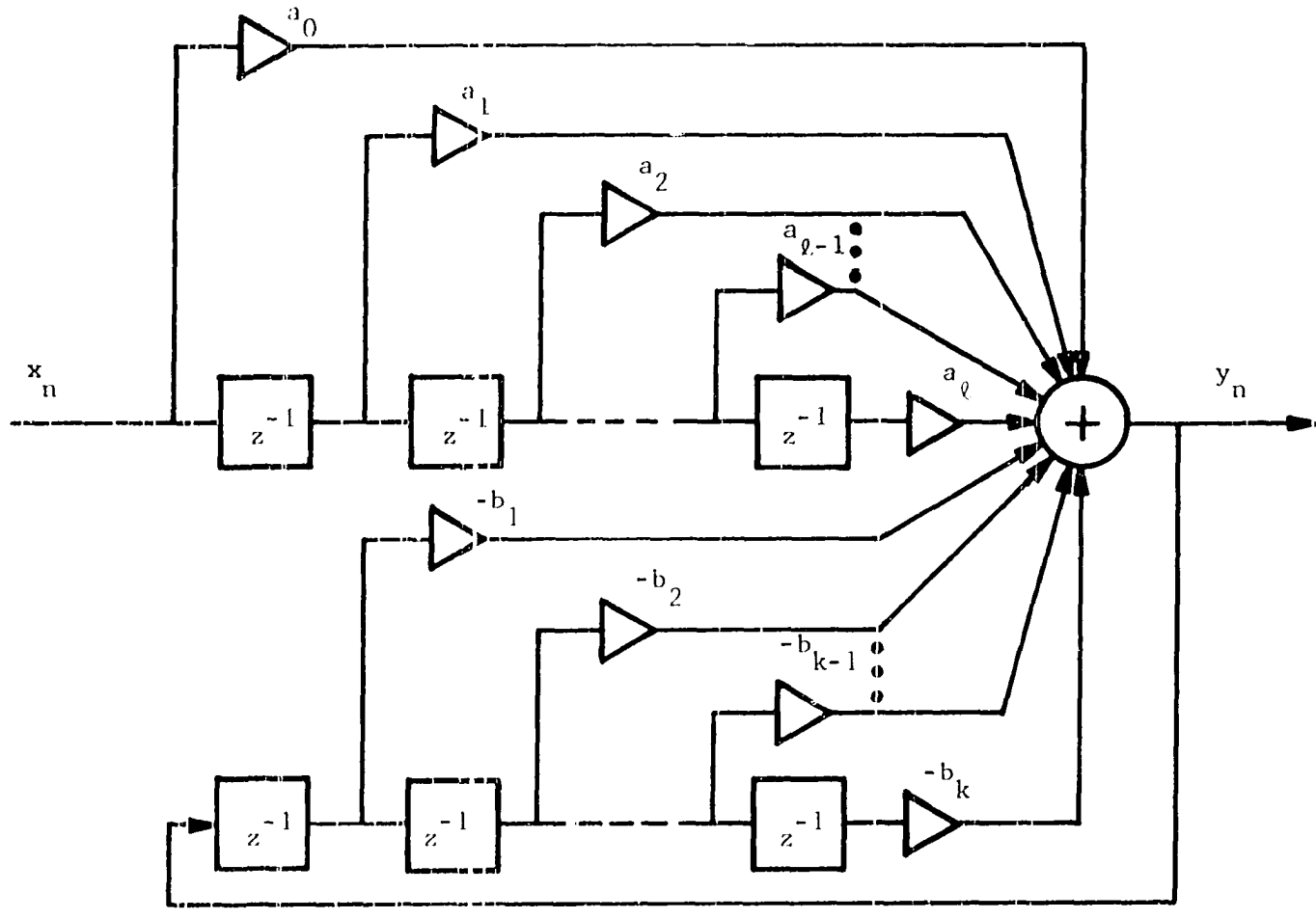


Figure 2. Direct recursive digital filter

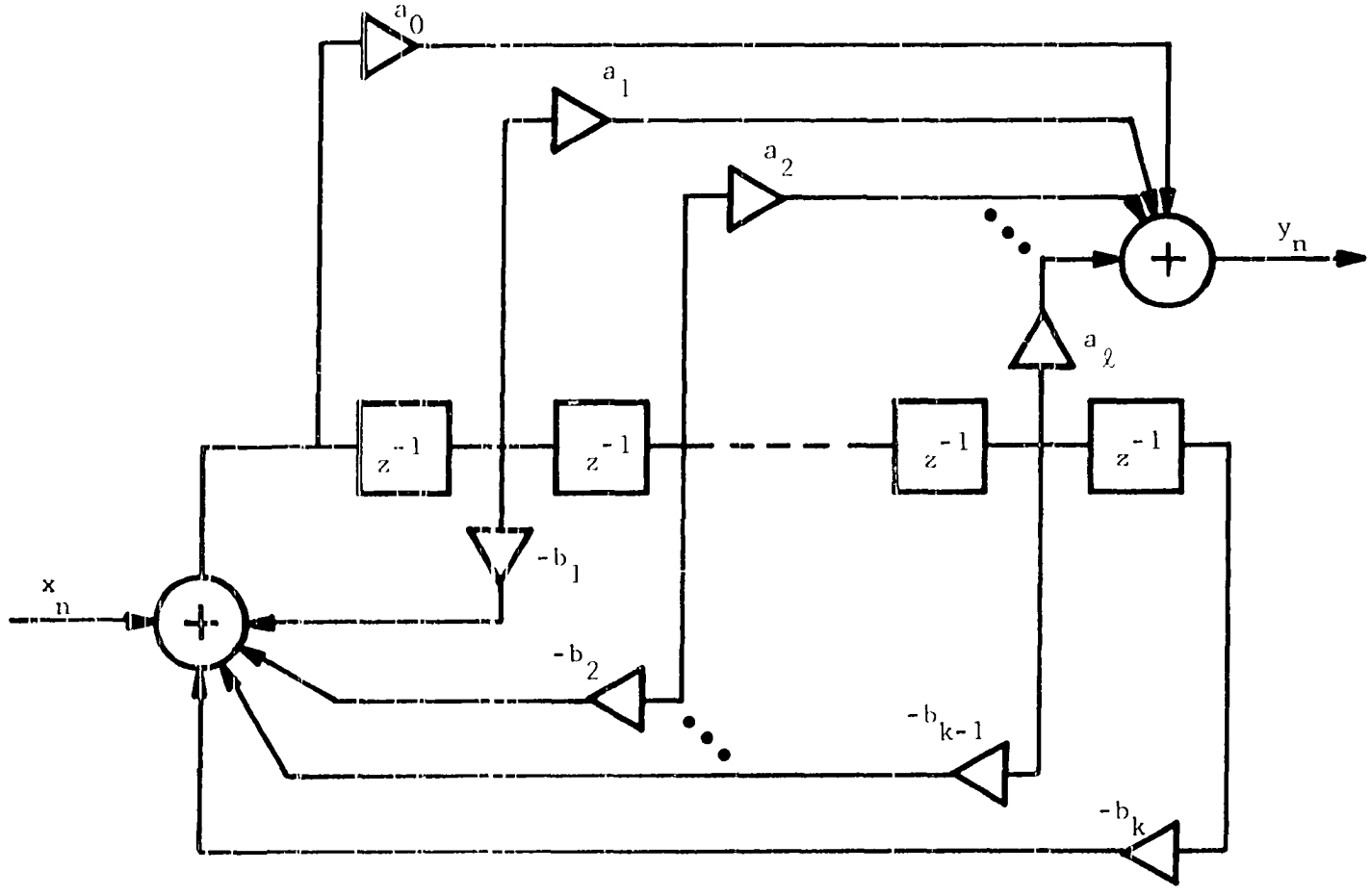


Figure 3. Direct canonic recursive digital filter

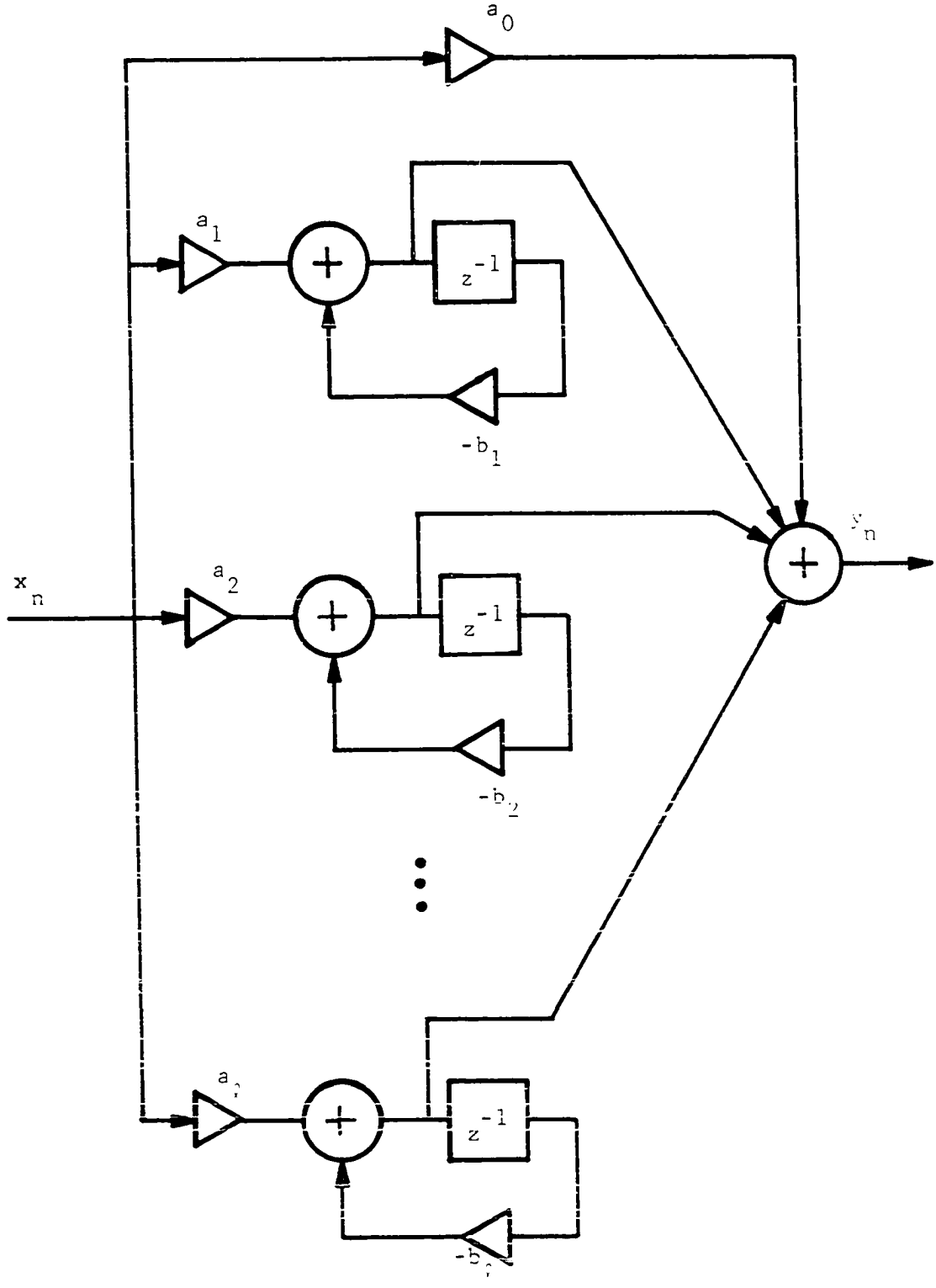


Figure 4. Parallel canonic recursive digital filter

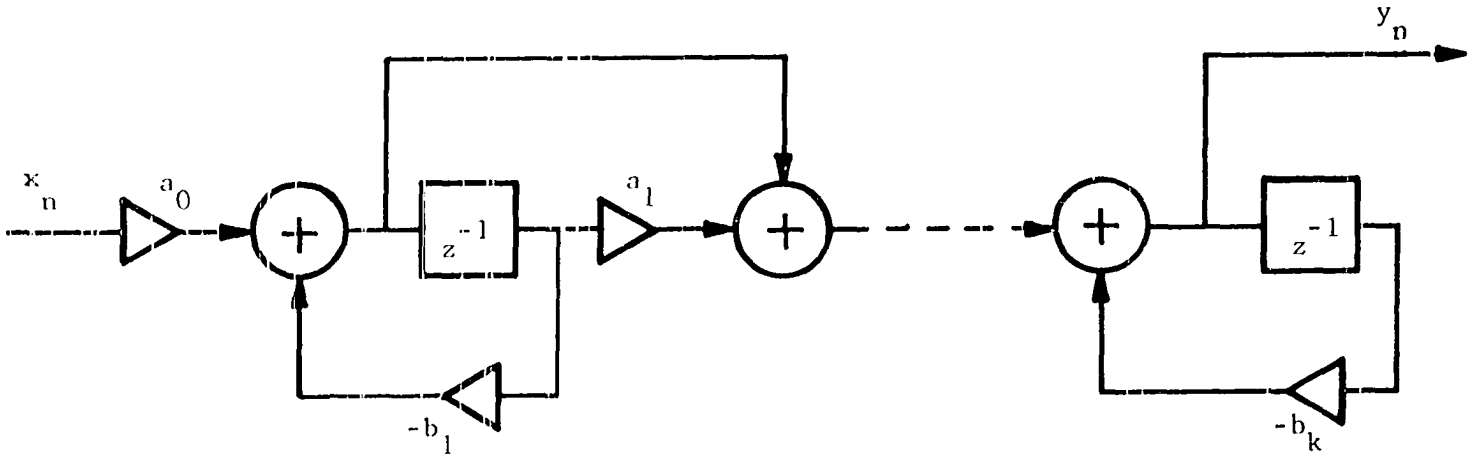


Figure 5. Cascade canonic recursive digital filter

One method used in this general approach is to expand the desired frequency response characteristic, $H(e^{j\omega T})$, into a Fourier series

$$H(e^{j\omega T}) = \sum_{k=-\infty}^{\infty} a_k e^{jk\omega T}$$

where

T = the sampling interval, and

a_k = the k^{th} Fourier coefficient.

Since $z = e^{j\omega T}$, then the frequency response characteristic may be represented as

$$H(z) = \sum_{k=-\infty}^{\infty} a_k z^{-k}.$$

To obtain a realizable filter, the Fourier series representation is truncated to

$$H_1(z) = \sum_{k-K}^{K} a_k z^{-k}$$

and the truncated expression is shifted to obtain the expression

$$\begin{aligned} H(z) &= z^{-K} H_1(z) \\ &= \sum_{k=-K}^{K} a_k z^{-k-K} \\ &= \sum_{\ell=0}^{2K} a_{\ell-K} z^{-\ell}. \end{aligned}$$

The filter is then implemented as an FIR digital filter.

Another method of designing a digital filter from a frequency domain point of view is to apply the time sampling equation

$$f(t) = \sum_{n=-\infty}^{\infty} f(nT) \frac{\sin\left[\frac{\pi}{T}(t-nT)\right]}{\frac{\pi}{T}(t-nT)}$$

to the desired frequency response characteristic, $H(e^{j\omega T})$, such that

$$H(e^{j\omega T}) = \sum_{n=-\infty}^{\infty} H(e^{jn\omega_0 T}) \frac{\sin\left[\frac{\pi}{\omega_0}(\omega-n\omega_0)\right]}{\frac{\pi}{\omega_0}(\omega-n\omega_0)} .$$

Recalling, again, that $z = e^{j\omega T}$, it is again possible to realize an FIR digital filter implementation.

Finally, if the desired filter response is known and is converted to a sampled time interval representation, h(nT), then a sampled data output, y(nT), can be realized by convoluting the sampled input with the sampled time interval filter response representation.

The convolution is more easily performed when the input and the filter transfer function are represented in the frequency domain. Since the input representation, x(nT), and the filter response, h(nT), are sequences of finite lengths, a discrete Fourier transform (DFT), an alternative Fourier transform representation, may be performed on these sequences. The convolution of x(nT) and h(nT) is carried out by taking the DFT of x(nT) and h(nT), multiplying X(f) and H(f), and then taking the inverse DFT of that product which yields the sampled filter response y(nT).

The discrete Fourier transform is defined as

$$X(mf) = \frac{1}{N} \sum_{n=0}^{N-1} x(nT)\, W^{-Mn} = \frac{1}{N} X(z)\Big|_{z = e^{jm\pi/N}}$$

where

$W = e^{j2\pi/N}$;

$f = \frac{1}{NT}$ = the discrete interval between frequency components;

$T = \frac{T_R}{N}$ = sampling interval;

T_R = duration of data.

The DFT is usually performed on sampled data using a fast Fourier transform (FFT) technique. The FFT, however, will not be discussed here.

Adaptive Digital Filtering

Widrow (16) provides an excellent description of adaptive digital filtering. In his introduction he refers to an adaptive filter as a self-designing or self-optimizing filter which bases its "design" on estimated statistical characteristics of the input and output signals of the filter. An adaptive filter may be implemented by using a recursive algorithm that automatically updates or modifies the filter system each time a new data sample is presented to the filter input. It is noted, however, that because the input and output signal statistical characteristics are estimated, errors will be present in the adaptive process and the filter will be less than optimal.

The basic filter described by Widrow, shown in Figure 6, is a non-recursive, finite impulse response (FIR), tapped delay line digital filter with variable gains or weights. Since these gains or weights determine the impulse response of the filter, the adaptation algorithm will seek to optimize the filter impulse response by adjusting these weights or gains.

Two separate processes take place in the adaptive filter algorithm. During the adaptation process, the filter weights are adjusted. To do this, a desired response signal must be provided to the adaptive filter in addition to the usual filter input. The filter output is then compared to the desired response and an error or difference signal is calculated. This error signal is used to modify the weights of the filter. Widrow showed that if the filter input and output signals are statistically stationary, the error signal has a mean-square value which is a quadratic function of the filter weights.

The second adaptive filter process is the operating process where the filter output signal is formed by summing appropriately weighted delay line inputs. The weights used in this process are those developed in the adaptation process.

If the input signals are expressed in vector form,

$$\bar{X}^T(j) = [x_1(j), x_2(j), x_3(j), \ldots, x_n(j)],$$

and the adaptive filter system weights are also presented in vector form,

$$\bar{W}^T(j) = [w_1(j), w_2(j), w_3(j), \ldots, w_n(j)],$$

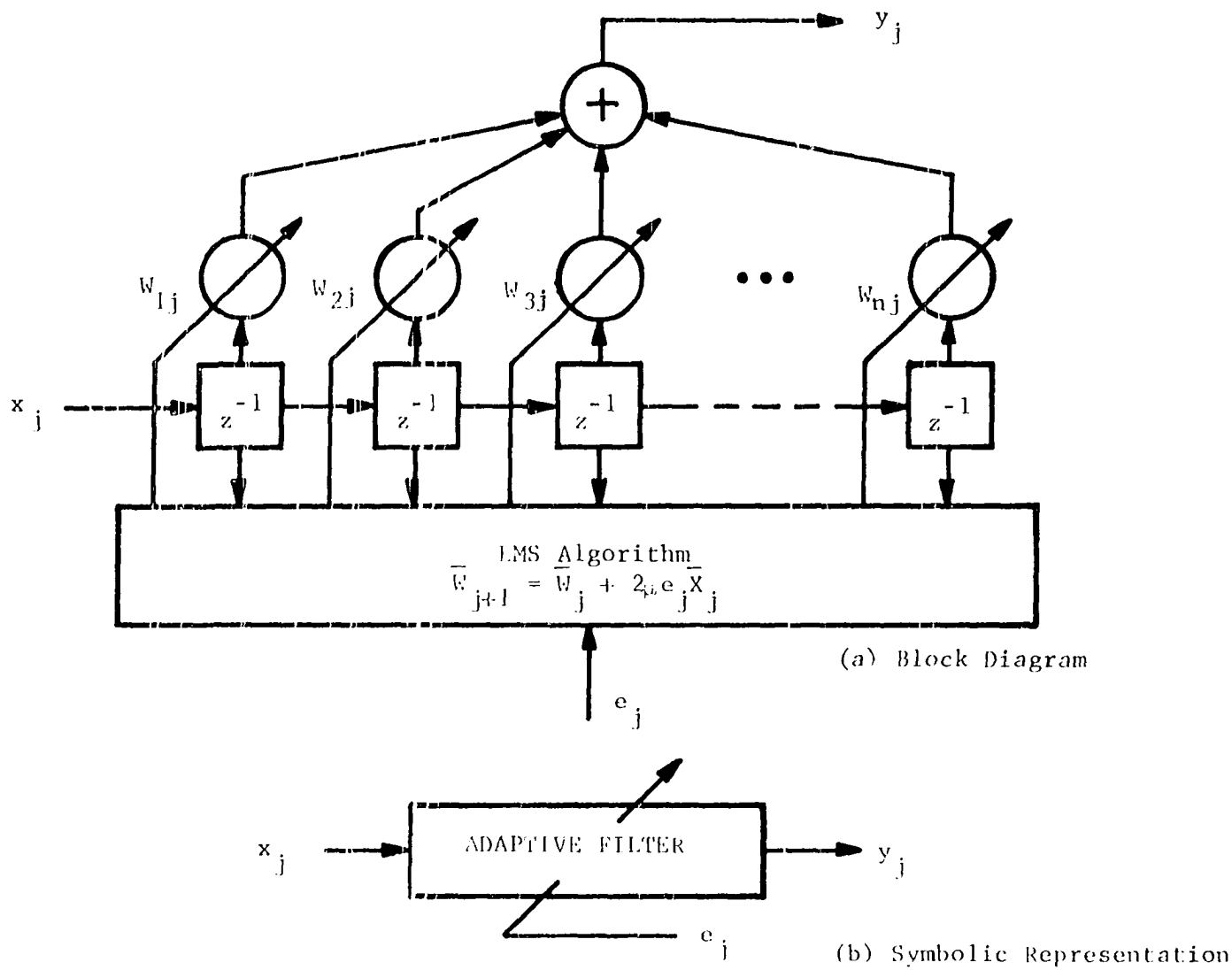


Figure 6. LMS adaptive filter

then the filter output may be expressed

$$y(j) = \sum_{\ell=1}^{n} \omega_\ell(j) x_\ell(j) = \overline{W}^T(j)\ \overline{X}(j) = \overline{X}^T(j)\ \overline{W}(j).$$

The error value is found by subtracting the desired output, d(j), from the filter output

$$e(j) = d(j) - y(j) = d(j) - \overline{W}^T(j)\ \overline{X}(j).$$

The mean-square error value

$$E[e^2(j)] = d^2(j) - 2\overline{\varphi}(x,d)\ \overline{W}(j) + \overline{W}^T(j)\overline{\varphi}(x,x)\ \overline{W}(j)$$

where

$$\overline{\varphi}(x,d) = E\begin{bmatrix} x_1(j)\ d(j) \\ x_2(j)\ d(j) \\ \cdots \\ x_n(j)\ d(j) \end{bmatrix}$$

and

$$\overline{\varphi}(x,x) = E\begin{bmatrix} x_1(j)\ x_1(j) & x_1(j)\ x_2(j) & \cdots & \cdots \\ x_2(j)\ x_1(j) & x_2(j)\ x_2(j) & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & x_n(j)\ x_n(j) \end{bmatrix}$$

The task now is to minimize the mean-square error value by selecting the appropriate weights.

To accomplish this minimization task, Widrow applies the method of steepest descent. This method involves taking an initial guess of the weight vector, taking the gradient of the mean-square error with respect

to the weight vector, changing the weight vector in the opposite direction of the gradient, and finally picking a new weight vector guess. This procedure can be shown as

$$\bar{W}(j+1) = \bar{W}(j) + k_s \bar{\nabla}[e^2(j)]$$

where k_s is a damping or convergence coefficient.

The disadvantage of using the method of steepest descent is that evaluation of $\bar{\Phi}(x,d)$ and $\bar{\Phi}(x,x)$ must be made for each calculation. The number of operations necessary for calculating $\bar{\Phi}(x,d)$ and $\bar{\Phi}(x,x)$ make this particular approach generally impractical when high input rates are to be implemented.

Widrow continues by describing an algorithm that he and M. E. Hoff, Jr. developed. This algorithm is based on the method of steepest descent and is called the least mean square or LMS algorithm. This algorithm does not require correlation function measurement, matrix inversion, squaring, averaging, or differentiation. The algorithm uses the basic steepest descent expression

$$\bar{W}(j+1) = \bar{W}(j) + k_s \bar{\nabla}[e^2(j)],$$

but calculates an estimate of the gradient of the mean-square error. In this case the estimated gradient is a single time sample of the squared error

$$\hat{\bar{\nabla}}[e^2(j)] = \bar{\nabla}[e^2(j)] = 2e(j)\bar{\nabla}[e(j)]$$

where

$$\bar{\nabla}[e(j)] = \bar{\nabla}[d(j) - \bar{W}^T(j)\,\bar{X}(j)] = -\bar{X}(j).$$

The weight iteration equation now becomes

$$\bar{W}(j+1) = \bar{W}(j) - 2k_s e(j)\,\bar{X}(j).$$

Widrow, et al. (17) presented an extensive discussion of adaptive noise cancellation using adaptive filtering. The basic technique involves subtracting an adaptive filtered reference noise source from a signal corrupted with noise. Further discussion is given to more specific applications of adaptive noise cancellation techniques. The applications cover cancelling noise and undesired signals in electrocardiography cancelling noise in speech signals, and cancelling antenna sidelobe interference.

Widrow, et al. (19) later discussed the performance characteristics of the LMS adaptive filter. In this discussion a plot of the mean-square error versus the number of filter iterations, called the "learning curve", was shown. Also, gradient and weight vector noise and the resulting adaptive filter misadjustment due gradient noise were discussed. The filter misadjustment was defined as

$$M \triangleq \frac{\text{average excess mean-square error}}{\text{minimum mean-square error}}$$

This misadjustment can also be shown to be

$$M = \frac{n}{4\tau}$$

where n = the number of filter weights, and

τ = the time constant of the learning curve.

Additional attention is given to the response of the LMS adaptive filter to nonstationary inputs.

Feintuch (4) has presented what he refers to as a recursive least mean square adaptive digital filter. This representation is shown in Figure 7. Using the recursive filter input-output relationship

$$y(n) = \sum_{k=0}^{N_F} a_k x(n-k) + \sum_{k=1}^{N_B} b_k y(n-k)$$

where

a_k represents a feed-forward coefficient;

b_k represents a feedback coefficient;

N_F is the number of feed-forward coefficients; and

N_B is the number of feedback coefficients.

This relationship is also expressed in vector notation as

$$y(n) = \bar{A}^T \bar{X}(n) + \bar{B}^T \bar{Y}(n)$$

where

$$\bar{A}^T = [a_0, a_1, \ldots, a_{N_F}];$$

$$\bar{B}^T = [b_1, b_2, \ldots, b_{N_B}];$$

$$\bar{X}(n) = [x(n), x(n-1), \ldots, x(n-N_F)];$$

and

$$\bar{Y}(n) = [y(n-1), \ldots, y(n-N_B)].$$

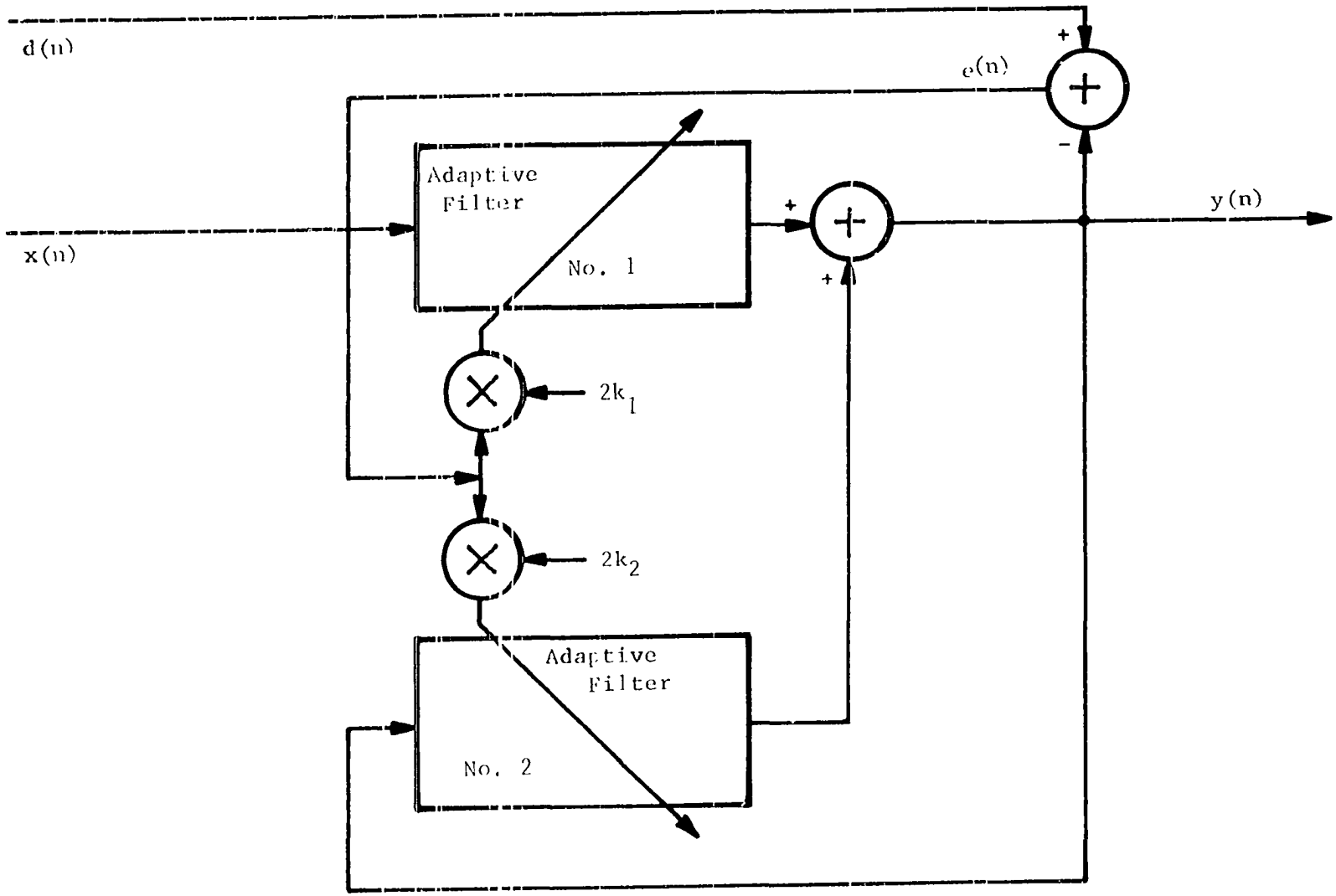


Figure 7. Recursive LMS adaptive algorithm

Feintuch has treated the recursive filter as two transversal filters and his application of the LMS algorithm to each filter is identical to its application to a single non-recursive filter. The results of this application are as follows:

$$\bar{A}(n+1) = \bar{A}(n) - 2k_1 e(n)\bar{X}(n)$$

and

$$\bar{B}(n+1) = \bar{B}(n) - 2k_2 e(n)\bar{Y}(n)$$

where k_1 and k_2 are the convergence coefficients for the two non-recursive filters.

Johnson et al. (10) have challenged that the adaptive recursive filter presented by Feintuch does not, in general, minimize the mean-square error. Johnson et al. (10) show the analysis of a single pole adaptive recursive filter modeling a two pole filter using Feintuch's algorithm. The analysis by Johnson et al. (10) concludes that the Feintuch algorithm does not necessarily converge to a zero gradient point and in a second example involving the adaptive modeling of a two pole recursive filter show that the algorithm actually drives the mean-square error away from the minimum.

Widrow et al. (18) also refute Feintuch's claim that the recursive filter minimizes the mean-square error in general. They do concede, however, that in some cases Feintuch's filter does minimize the mean-square error, but in other cases the algorithm will converge to non-minimum mean-square error values.

Feintuch, in responding to these challenges, maintains that his algorithm does indeed represent an LMS adaptive recursive digital filter and, while many portions of this filter's operation are as yet unclear, the results of several simulation examples presented show that an adaptive recursive filter is technically feasible.

Parikh et al. (13) present an adaptive recursive filter algorithm which updates the filter coefficients by implementing Stearn's algorithm. The representation of this filter is similar to that shown in Figure 7. Consider an adaptive filter transfer function

$$H_a(z_1k) = \frac{a(0,k) + a(1,k)\ z^{-1} + \ldots + a(M,k)\ z^{-M}}{1-b(1,k)\ z^{-1} - b(2,k)\ z^{-2} - \ldots - b(N,k)\ z^{-N}}$$

where k is the time index and a(i,k) and b(j,k) denote the filter coefficients. The corresponding input-output relationship is then expressed in vector form as

$$g(k) = \bar{\alpha}_k^T \bar{\varrho}_k$$

where g(k) is the output at time k,

$$\bar{\alpha}_k^T = [a(0,k)\ a(1,k)\ \ldots\ a(M,k)\ b(1,k)\ b(2,k)\ \ldots\ b(N,k)]$$

is the transpose of the weight vector, and

$$\bar{\varrho}_k^T = [x(k)\ x(k-1)\ \ldots\ x(k-M)\ g(k-1)\ g(k-2)\ \ldots\ g(k-N)],$$

is the transpose of the input vector.

Taking the gradient of g(k) with respect to the weight vector yields

$$\bar{\nabla} g(k) = \bar{\varrho}_k + \sum_{p=1}^{N} b(p,k)\ \bar{\nabla}_\alpha g(k-p)$$

where $\bar{\nabla}_{\alpha}\, g(k-p)$ denotes the gradient of $g(k-p)$ with respect to α. This expression can be rewritten to show the recursive computation of the gradient

$$\bar{\lambda}_k = \bar{\varepsilon}_k + \sum_{p=1}^{N} b(p,k)\, \bar{\lambda}_{k-p}$$

where $\bar{\lambda}_k$ is the recursive realization of $\bar{\nabla}_{\alpha}\, g(k-p)$. The recursive updating of the filter coefficients now becomes

$$a(i,k) = a(i,k-1) + u\, e(k)\lambda_k(i),\ i = 0,\ 1,\ \ldots,\ M$$

and

$$b(j,k) = b(j,k-1) + v\, e(k)\lambda_k(i),\ j = 1,\ 2,\ \ldots,\ N$$

where u and v are convergence parameters,

$$e(k) = y(k) - \bar{\alpha}^{T}_{k-1}\, \bar{\varepsilon}_k$$

$$= \text{error at time } k,$$

and $y(k) = \text{corresponding desired output.}$

These expressions can be further reduced by approximating $\bar{\lambda}_k \times \bar{\varepsilon}_k$ so that

$$a(i,k) = a(i,k-1) + u\, e(k)\, x(k-i),\ i = 0,\ 1,\ \ldots,\ M$$

and

$$b(j,k) = b(j,k-1) + v\, e(k)\, g(k-j),\ j = 1,\ 2,\ \ldots,\ N$$

which Parikh and Ahmed point out is really Feintuch's algorithm.

Using the example that Johnson et al. used in challenging Feintuch's algorithm, Parikh and Ahmed show that the Stearn's algorithm implementation of the adaptive recursive filter problem does converge to either a global or a local minimal of the mean-squared error. Johnson

and Larimore, in responding to Parikh and Ahmed, point out that the improved performance of this algorithm is obtained, however, at the expense of additional complexity of the adaptive algorithm.

Microprocessors in Digital Filtering

Jenkins (9) has presented several ideas on microprocessor-based adaptive digital filter architectures. Such a basic structure is shown in Figure 8. The systems proposed incorporate residue number systems and multiple/parallel processor architectures. A separate processor is used for each prime modulus in the residue number system. However, the sizes of the different moduli are constrained by the accumulator size of the microprocessor used. Each processor then performs a separate LMS adaptive digital filtering function described earlier, but in its own prime modulus number system. A scalar processor is used to scale the output of each parallel processor and combine these outputs into a composite value. (It is noted that the order of the filter is still a constraint for the residue number system implemented adaptive digital filter.)

Since all of the digits of the residue number system (that is, the outputs of each parallel processor) are of equal significance, a failure of a single parallel processor will not affect the other parallel processors' operation. Therefore, the scalar processor could check for parallel processor errors. If one is detected, the faulty processor's output is ignored, the scalar function manipulated to compensate for the

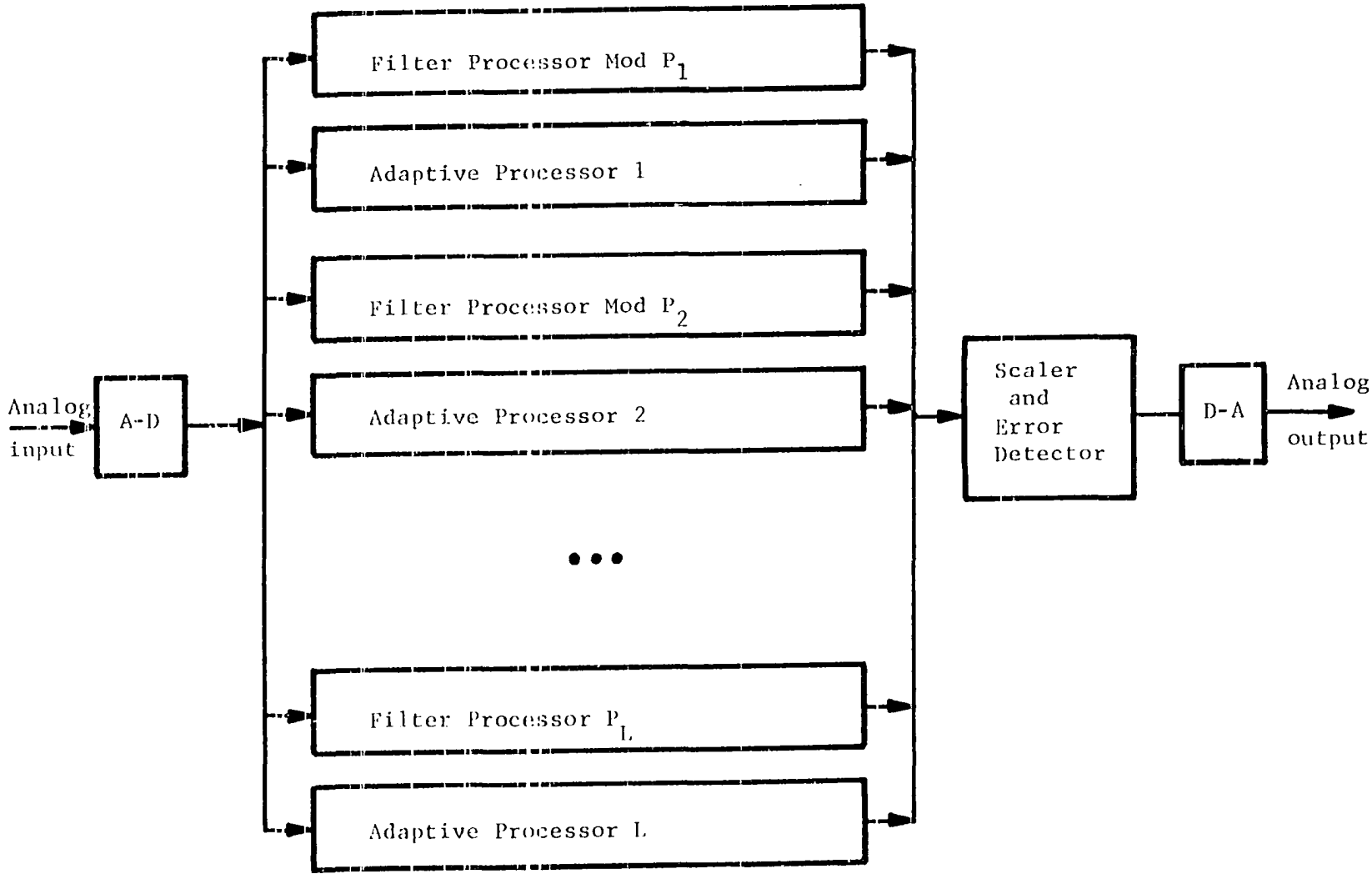


Figure 8. Fault tolerant residue number system multiprocessor LMS adaptive digital filter

lost parallel output, and the filter can continue to operate, but with diminished performance.

Further increases in filter performance are possible by incorporating a separate processor for each modulus processor. This additional processor would perform the adaptive functions while the original modulus processor would perform only the filter function for its assigned modulus.

Soderstrand (14) proposed a three radix multiple microprocessor residue number system digital filter using the COSMAS CDP1802 microprocessor. While this system is similar to the concept proposed by Jenkins, the moduli processors do not perform filter functions themselves, but rather direct the data to special purpose hardware compenents which perform the filter functions.

Soderstrand claims that data rates of 10,000 - 20,000 samples per second are possible for non-recursive low order digital filters using this system. However, as might be expected, the data rate is substantially reduced when recursive, higher order, or adaptive algorithms are implemented.

FILTER SPECIFICATION

Specification Derivation

The original goal of this project was to develop a filter which might be used to aid in the determination of heart rate information for a wide variety of animals. While the filtering of heart rate information will not be demonstrated here, the filter specification developed reflects many of the basic requirements necessitated by the original goal. Some of the requirements for a filter of this nature were not known and, as a result, certain points in the specification were either selected arbitrarily or simply left unspecified.

To adequately sample a human electrocardiograph signal, Wartak (15) recommends a sampling rate of 200-250 samples per second. For detection of the human QRS heart wave using digital differentiation, Holsinger (5) et al., recommend a sampling rate of 250 samples per second. The human heart rate can generally range from 0 to as high as 180 beats per minute. Generally, small animals have higher heart rates than humans. A possible heart rate range for small animals might be between 0 and 300 beats per minute which corresponds to a frequency range of 0 to 5 Hz. Wartak (15) claims that the significant heart signal components are distributed between the fundamental frequency and the fortieth harmonic.

Generally, the third harmonic of the heart signal has more amplitude than the other components and it is this third harmonic component which can be detected for determining the heart rate.

A major source of contamination of the heart signal is muscle noise generated by the animal being monitored. Muscle noise is generated by animal muscle operation (other than the heart muscle) during periods of physical activity and it can have a magnitude greater than that of the heart signal. It is therefore necessary to bandpass filter the heart rate signal to attenuate this resultant muscle noise. Since, as stated before, the heart rate for the same animal can vary, it is desirable to have the bandpass filter track the heart signal frequency. Adaptive digital filtering techniques were studied and found to be possible approaches to bandpass filter center frequency tracking.

With these requirements in mind and after including a margin of safety, the following specification statement was the result.

Specification Statement

The filter shall be a fourth order bandpass digital filter which will be implemented in real time utilizing microprocessor control. The digital filter shall have a fixed bandwidth while the center frequency will be allowed to vary according to the frequency of the signal being filtered. The frequency band of interest or baseband shall be from 0 to 180 Hz which will require a Nyquist rate sampling frequency of at least 360 Hz.

SYSTEM DESIGN

The filter development phase involved deriving the total algorithm from the specification and then defining the hardware and software systems to implement the total filter algorithm. The overall development of the filter was accomplished in the following manner.

1. The fourth order digital bandpass filter algorithm was developed and verified;
2. Those fourth order digital bandpass filter parameters which are center frequency dependent were noted;
3. An adaptive algorithm was developed which exploits the center frequency dependent digital filter parameters;
4. The hardware and software functions of the adaptive filter system were determined; and
5. The filter system was implemented in hardware and software.

The following is a detailed discussion of these steps.

Basic Filter Design

The basic nonadaptive fourth order filter was chosen to be a Butterworth bandpass filter. While the Butterworth filter is characterized by a maximally flat passband, the choise of this type of filter was strictly arbitrary. A Chebyshev or elliptic type of filter could also have been selected and the procedure for the conversion to a digital implementation would have been the same.

The approach to developing a digital version of this filter was:

1. Start with the s-domain expression for a second order Butterworth lowpass filter;
2. Develop and use a substitution factor to transform the s-domain lowpass filter expression directly into an expression for a fourth order z-domain Butterworth bandpass filter; and
3. From the fourth order z-domain bandpass filter expression, derive the digital filter diagram.

The normalized s-domain transfer function for a second order lowpass Butterworth filter,

$$H(s) = \frac{1}{s^2 + \sqrt{2}\, s + 1}$$

was derived from the generalized transfer function given by Antoniou (1).

A substitution factor to convert the s-domain lowpass filter expression to a z-domain bandpass filter expression was then developed. This factor is based on the substitution factor given by Childers and Durling (2) which is used to convert an s-domain lowpass filter to an s-domain bandpass filter

$$s_{LP} = \frac{s_{BP}^2 - \omega_c^2}{(\omega_2 - \omega_1)\, s_{BP}}$$

where

s_{LP} = lowpass s operator;

s_{BP} = bandpass s operator;

ω_2 = bandpass upper cutoff frequency;

ω_1 = bandpass lower cutoff frequency; and

$\omega_c = \sqrt{\omega_2\omega_1}$ = bandpass filter center frequency.

The bilinear transformation was selected to convert the expression from the s-domain to the z-domain because of its freedom from aliasing or the extension of the amplitude response beyond the limits of the baseband. The bilinear transformation is performed by substituting

$$s = \frac{2}{T}\frac{z - 1}{z + 1}$$

where T is the sampling interval, into the s-domain continuous transfer function expression. However, when the bilinear transformation is used it causes a warping or shifting of the frequency between the analog and digital transfer functions. The relationship between the desired and warped frequencies is

$$\omega_s = \frac{2}{T}\tan^{-1}\frac{\omega_0 T}{2}$$

where

ω_s = warped frequency;

ω_0 = desired frequency; and

T = sampling interval.

Therefore, by prewarping the frequency of the s-domain lowpass to bandpass substitution factor, that is, substituting this expression into the s-domain lowpass to bandpass substitution factor, and, finally, substituting the bilinear transformation expression, the lowpass continuous filter to bandpass digital filter substitution factor expression becomes

$$s_{LP} = \frac{\left(\frac{1 - z^{-1}}{1 + z^{-1}}\right)^2 + \tan\frac{\omega_2 T}{2}\ \tan\frac{\omega_1 T}{2}}{\left(\frac{1 - z^{-1}}{1 + z^{-1}}\right)^2 \tan\frac{\omega_2 T}{2}\ \tan\frac{\omega_1 T}{2}}$$

$$= \frac{(1+\tan\frac{\omega_2 T}{2}\tan\frac{\omega_1 T}{2}) - 2z^{-1}(1-\tan\frac{\omega_2 T}{2}\tan\frac{\omega_1 T}{2}) + z^{-2}(1+\tan\frac{\omega_2 T}{2}\tan\frac{1 T}{2})}{(1 - z^{-2})(\tan\frac{\omega_2 T}{2} - \tan\frac{\omega_1 T}{2})}$$

$$= \frac{1}{(1 - z^{-2})}\left[\frac{1}{\tan(\frac{\omega_2 - \omega_1}{2})T} - \frac{2z^{-1}(1-\tan\frac{\omega_2 T}{2}\tan\frac{\omega_1 T}{2})}{\tan\frac{\omega_2 T}{2} - \tan\frac{\omega_1 T}{2}} + \frac{z^{-2}}{\tan(\frac{\omega_2-\omega_1}{2})T}\right]$$

$$= \frac{\cot(\frac{\omega_2 - \omega_1}{2})T}{1 - z^{-2}}\left[1 - 2\frac{\cos(\frac{\omega_2+\omega_1}{2})T}{\cos(\frac{\omega_2-\omega_1}{2})T}\ z^{-1} + z^{-2}\right]$$

To further simplify this expression, let

$$k = \cot(\frac{\omega_2-\omega_1}{2})T$$

and

$$\alpha = \frac{\cos(\frac{\omega_2+\omega_1}{2})T}{\cos(\frac{\omega_2-\omega_1}{2})T}$$

so that the final substitution expression becomes

$$s_{LP} = k\ \frac{1 - 2\alpha\ z^{-1} + z^{-2}}{1 - z^{-2}}$$

which is the expression shown by Constantinides (3) for converting a lowpass continuous filter having a cut-off frequency of 1 radian per second to bandpass digital filter.

Next, to convert the s-domain lowpass filter expression to a z-domain bandpass expression, substitute the expression

$$s_{LP} = k \frac{1-2\alpha z^{-1} + z^{-2}}{1 - z^{-2}}$$

into the continuous s-domain lowpass filter expression

$$H(s) = \frac{1}{s^2 + \sqrt{2}s + 1} .$$

Hence,

$$H(z) = \frac{1}{k^2\left(\frac{1-2\alpha z^{-1}+z^{-2}}{1 - z^{-2}}\right)^2 + \sqrt{2}k\left(\frac{1-2\alpha z^{-1}+z^{-2}}{1 - z^{-2}}\right) + 1}$$

$$= \frac{\frac{1}{k^2+\sqrt{2}k+1} - \frac{2z^{-2}}{k^2+\sqrt{2}k+1} + \frac{z^{-4}}{k^2+\sqrt{2}k+1}}{1 - \frac{2\alpha k(-2k+\sqrt{2})}{k^2+\sqrt{2}k+1} z^{-1} + \frac{2(2\alpha^2k^2+k^2+1)}{k^2+\sqrt{2}k+1} z^{-2} + \frac{2\alpha k(-2k+\sqrt{2})}{k^2+\sqrt{2}k+1} z^{-3}}$$

$$+ \frac{k^2 - \sqrt{2}k+1}{k^2 + \sqrt{2}k+1} z^{-4}$$

Using this expression for H(z), a diagram of this bandpass filter can be realized in a direct canonic form as shown in Figure 9.

It must be mentioned that a complex input signal will be contaminated by high frequency components from above the highest frequency of interest or upper baseband frequency. These high frequency components will cause aliasing if they are allowed in the digital filter. For this

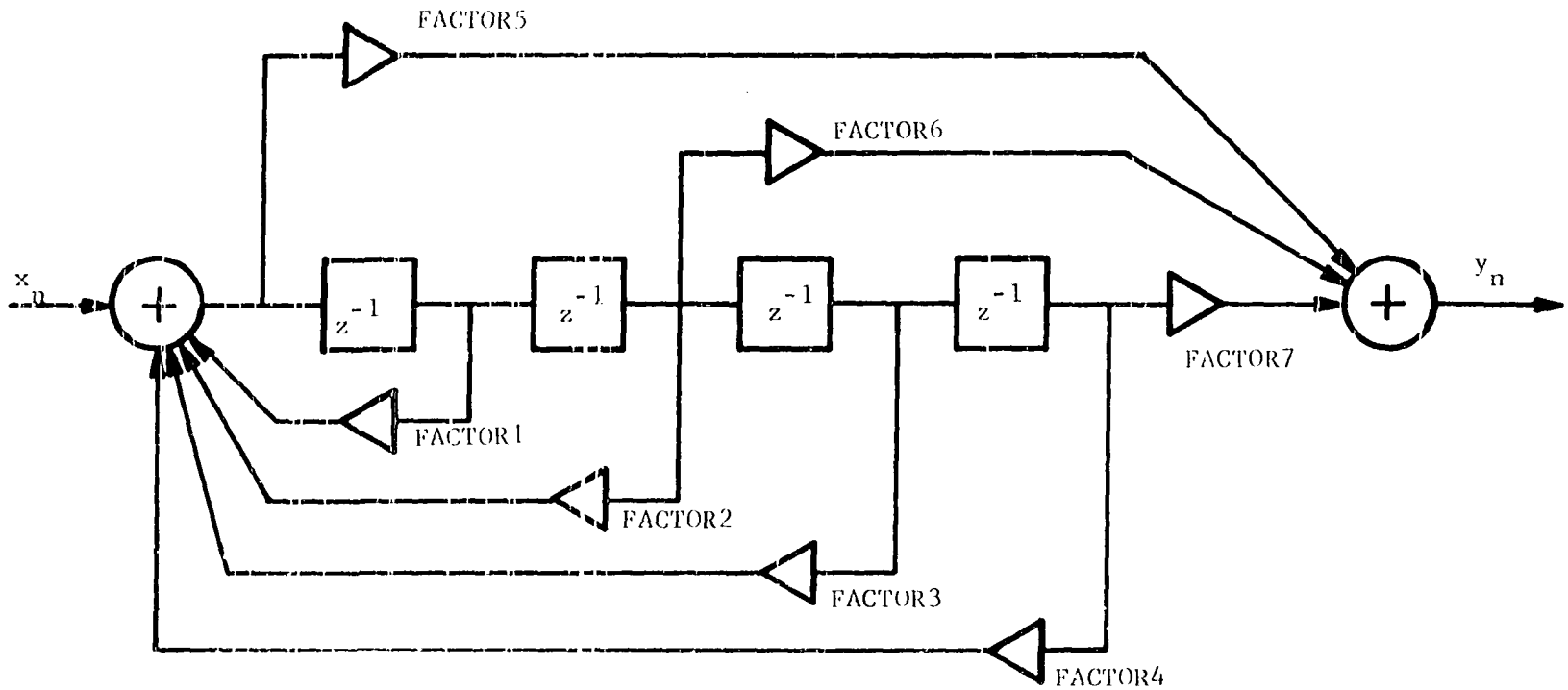


Figure 9. Fourth order bandpass recursive digital filter

reason a simple single pole lowpass filter with a cutoff frequency at the highest frequency of interest is usually placed prior to the input of the digital filter.

In like manner, a similar lowpass filter is usually placed in the output of the digital filter. This lowpass filter removes the high frequency components of the discrete nonlinear filter output waveform.

These lowpass filters are not included in this filter system implementation shown in Figure 9 since this system is being tested with only a sine wave input signal. However, if complex input signals were to be included, these lowpass filters would need to be a part of the overall filter system.

The expressions for the weighting factors or coefficients are obtained directly from the bandpass filter transfer function expressed in the z-domain; i.e.

$$\text{FACTOR1} = \frac{2\alpha k\ (2k + \sqrt{2})}{k^2 + \sqrt{2}k + 1}$$

$$\text{FACTOR2} = \frac{-2(2\alpha k^2 + k^2 - 1)}{k^2 + \sqrt{2}k + 1}$$

$$\text{FACTOR3} = \frac{2\alpha k\ (2k - \sqrt{2})}{k^2 + \sqrt{2}k + 1}$$

$$\text{FACTOR4} = -\frac{k^2 - \sqrt{2}k + 1}{k^2 + \sqrt{2}k + 1}$$

$$\text{FACTOR5} = \frac{1}{k^2 + \sqrt{2}k + 1}$$

$$\text{FACTOR6} = -\frac{2}{k^2 + \sqrt{2}k + 1}$$

$$\text{FACTOR7} = \frac{1}{k^2 + \sqrt{2}k + 1}$$

Using the above expressions for the weight factors, three simulation programs were developed and written in PL/1 to verify this nonadaptive filter algorithm. The listings of these programs are presented in Appendix 1.

The first program, called SCAN, accepts the filter bandwidth and the filter sampling rate and calculates the seven weight factors or parameters of this filter for center frequencies 2 Hz apart within the frequency band of interest or baseband for a fixed filter bandwidth. (The program could be modified to display these parameters for other center frequency separations.) These filter parameter values are presented in both decimal and hexadecimal forms.

The second program, called FILTER, calculates the filter output values for input frequencies in 10 Hz increments throughout the baseband. This program was used to verify that the filter algorithm would be realizable.

The third program, called RESPONS, calculates the amplitude frequency response of the filter with a fixed center frequency and filter

bandwidth. The input parameters for this program are the filter upper and lower cutoff frequencies and the upper baseband frequency. This program is used to verify that the response of this digital filter is indeed the response of a bandpass filter. The data generated by this program are used for comparison with the filter response data obtained from the actual digital filter system.

When the filter weight factor expressions were examined, it became obvious that only those weight factors which contain the component α are center frequency dependent. The component $k = \cot(\frac{\omega_2-\omega_1}{2})T$ is dependent on the filter bandwidth which, recall, for this application is fixed or constant. Therefore, only FACTOR1, FACTOR2, and FACTOR3 will be modified by the adaptive filter algorithm.

Having determined the center frequency dependent filter weight factors, the next step was to develop the adaptive filter algorithm. A diagram of the adaptive filter system is shown in Figure 10.

Adaptive Algorithm

The adaptive algorithm is basically a binary tree search of the frequency band of interest or baseband to find the bandpass digital filter configuration which matches or comes closest to matching its center frequency with the filter input signal frequency.

Assume that there is an input signal applied to the filter input whose frequency is somewhere within the baseband. Initially, the digital filter is configured with its center frequency at the center of the

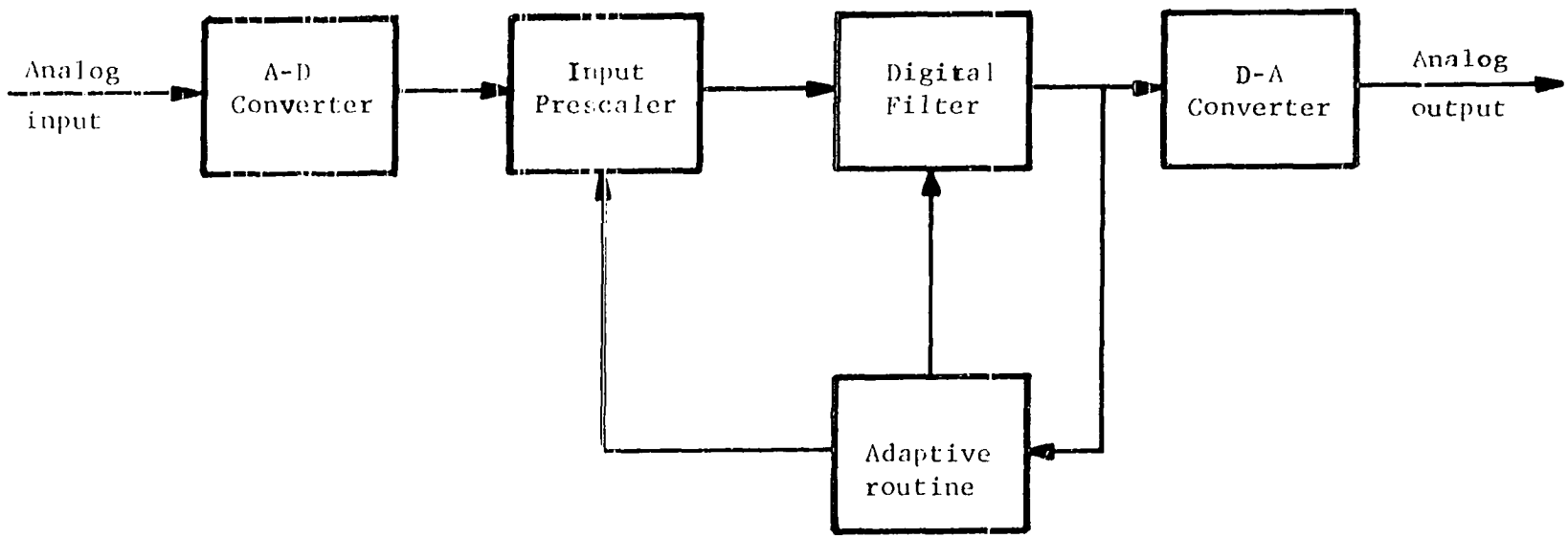


Figure 10. Adaptive filter block diagram

baseband, in this case 90 Hz. This frequency is termed the initial node frequency. If the output of the filter at the initial node frequency does not exceed or equal a predetermined threshold value, which indicates that the filter input signal frequency was not 90 Hz, the filter is reconfigured to have a center frequency, first at a frequency midway between the center and upper limit of the baseband, and second, at a frequency midway between the lower limit and center of the baseband. These frequencies are referred to as the right subnode frequency and the left subnode frequency, respectively. The peak outputs of the filter configurations centered on these subnode frequencies are then compared and the subnode frequency of the filter configuration with the greater peak output value is now considered to be the node frequency and the performances of the filter configured at the subnode frequencies of this new node frequency are considered.

This process repeats until the peak output value of the filter configured at a node or subnode frequency equals or exceeds the predetermined threshold or a node frequency is found which does not have any subnode frequencies which indicates that the bottom of the tree has been reached. Generally, the distance or bandwidth between subnode frequencies in succeeding sets or formal pairs is one half that of the pair immediately preceding it. Thus, assuming correct detections of the filter peak output values, each succeeding iteration of this adaptive algorithm should move the filter center frequency closer to the input signal frequency.

Hardware Development

The Intel 8080A was selected as the microprocessor to perform the digital filtering function. The 8080A is a parallel 8 data bit processor capable of accessing 65,536 bytes of memory as well as 256 discrete inputs and 256 discrete outputs. The 8080A includes a 16 bit stack pointer register, a 16 bit program counter, six 8 bit general purpose registers arranged into three register pairs, as well as an 8 bit accumulator register. The 8080A instruction set has a repertoire of 111 discrete instructions which are capable of memory and register manipulation, accumulator and stack operations, conditional and unconditional jump commands and subroutine calls. The 8080A can also handle vectored interrupts. The 8080A can be driven by a clock frequency of up to 2 MHz which allows for a machine cycle of 0.5 microseconds in duration (8).

While the Intel 8080A is a very capable microprocessor, it is not the best suited for this type of application. However, the 8080A was selected for this task because of the immediate availability of the Intel Intellec 800 Microcomputer Development System (MDS). The MDS is a powerful microprocessor system development tool on which to develop the system software and incorporate and test the software.

A block diagram of the MDS is shown in Figure 11. The MDS is itself an 8080A based microcomputer system. The version of the MDS available for this project is driven by a 2 MHz clock and has 64K bytes (65,536 bytes, to be exact) of random access memory (RAM) available as well as dual diskette storage.

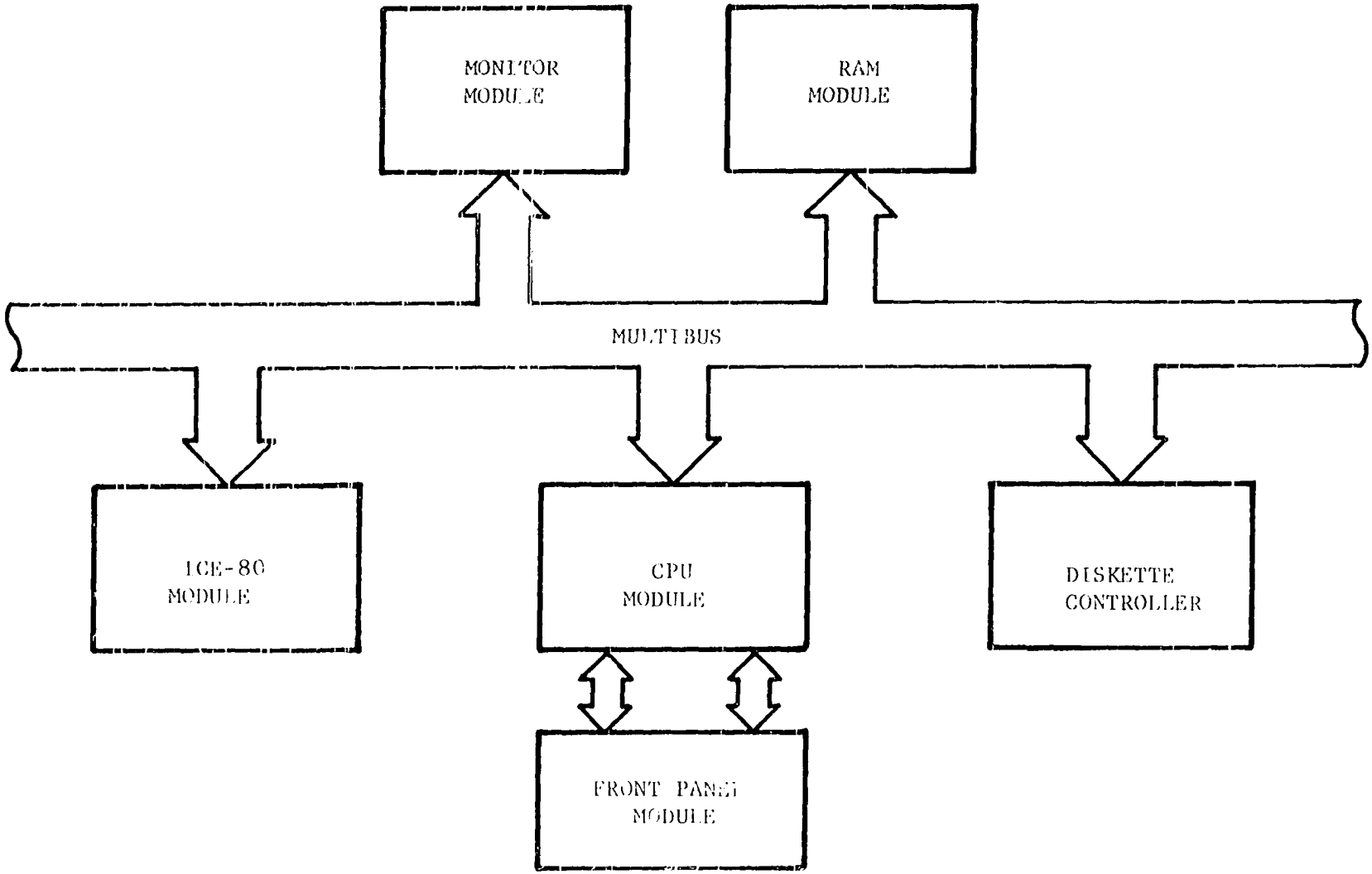


Figure 11. Intel Intellec 80C MDS block diagram

All of the MDS modules shown in Figure 11 are interconnected by the Intel MULTIBUS. The MULTIBUS contains 20 address lines, 16 bi-directional data lines, interrupt control lines as well as bus control lines, timing lines, and power lines (7).

The overall MDS is controlled by the Intel Systems Implementation Supervisor (ISIS-II) diskette based operating system which resides on diskette. The ISIS-II operating system controls access to a text editor, a resident 8080 assembler, a "linker" routine for linking assembled program modules together, a "locate" routine to locate assembled modules almost anywhere in the MDS memory, an in-circuit emulator (ICE-80) for testing software and incorporating the software with the hardware, and a read-only memory (ROM) based monitor program for use in running assembled programs. ISIS-II also makes the diskette an input-output device and provides for diskette file generation and management (6).

It was also decided to utilize the Analog Devices RTI-1200 A/D - D/A real time interface system since it was also readily available. A block diagram of the RTI-1200 is shown in Figure 12. The RTI-1200 system has 16 differential analog input channels and 2 differential analog output channels. The RTI-1200 converts each analog input into 12 bit binary form. Thus, 4096 quantization levels are possible which result in a 0.024% accuracy and a potential 72 db signal to noise ratio. Control signals and the 12 bit data information pass between the RTI-1200 and its controlling microcomputer, the 8080A in this case, via the Intel

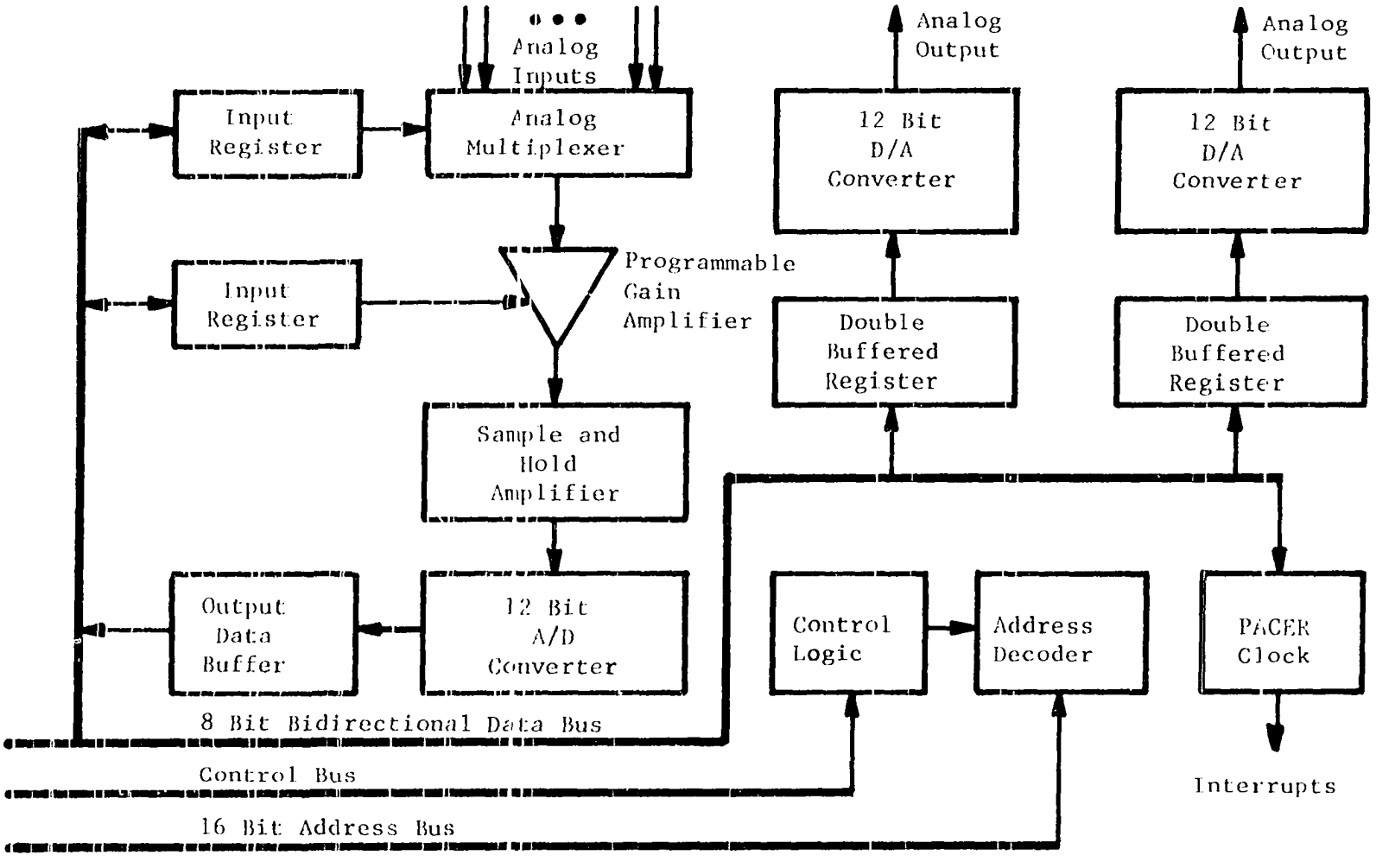


Figure 12. RTI-1200 block diagram

MULTIBUS. The RTI-1200 also includes a "PACER" clock system which can be used to interrupt the controlling microcomputer. One portion of the "PACER" system is a settable R-C clock.

Hardware Multiplication

One obvious conclusion from studying the nonadaptive digital filter algorithm diagram in Figure 9 is that seven multiplications are required to perform a single iteration of this filter algorithm. Therefore, the next logical task would be to develop a fast, efficient multiplication software routine in 8080A machine language. Since the digital input values from the RTI-1200 are 12 bits long and the 8080A accumulator is 8 bits long, the multiplication routine should be capable of multiplying a 12 bit multiplicand times an 8 bit multiplier.

Nine different multiplication routines were developed. These multiplication routines were variations of the shift-and-add, Booth, and table-look-up algorithms. All nine routines were compared using twelve different test cases. The results of this comparison showed that the fastest multiplication routine could, in most cases, complete seven multiplications within the specified filter sample interval. However, there was practically no time left to perform other operations, such as 16 bit additions and A-D and D-A converter control, which are necessary to the basic filter function in the remaining time of that interval. The other routines performed at even slower rates. It was obvious that these software multiplication routines were unacceptable when the adaptive routine was incorporated.

Several solutions to this problem were examined. It was decided to incorporate a hardware multiplication feature to the overall filter system rather than decrease the requirements of the specification. To this end a hardware multiplication board based on the TRW TDC1010J 16 bit x 16 bit multiplier-accumulator was designed, constructed, and tested. A detailed description of this hardware multiplier is given in Appendix 2.

While this hardware multiplier function is not a particularly efficient application of the TDC1010J multiplier-accumulator, its use does solve the multiplication speed problem and also offers the following advantages:

1. Since the TDC1010J multiplies two 16 bit numbers, the filter weight factor values can be increased to 16 bits in length;
2. The TDC1010J can add or subtract newly formed products in its own internal accumulator. Filter feedback and feed-forward values can thus be quickly and easily calculated using the hardware multiplier; and
3. All TDC1010J calculations can be done in two's complement notation.

The hardware multiplier board is also controlled by the filter system program. The data buffers and the multiplication overhead operations are directed by memory mapped operations from the 8080A microprocessor. These buffers and functions were memory mapped rather

accessed through input and output ports because, in this configuration, data values can be stored and fetched from the multiplier board using 8080A 16 bit data instructions rather than being restricted to 8 bit input/output port operations.

Also mounted on the multiplier board is a manual toggle switch which sets a software switch to signal the performance of the adaptive portion of the filter system program. This switch is not an integral portion of the total filter system, but has been incorporated to disable the adaptive portion of the filter system to allow for analysis of the modified filter weight factors.

The hardware multiplier board, like the RTI-1200, is also compatible with the Intel MULTIBUS which makes both boards plug-in compatible with the MDS. This greatly simplifies system development and testing.

Software Development

Since the "PACER" clock system was available with the RTI-1200, it was decided to use the settable R-C timer to interrupt the 8080A at the specification sampling rate of 360 Hz. When the microcomputer is interrupted, it will perform one complete iteration of the filter program. However, since the filter is to operate in real time, the complete filter program must be completed within a single sampling interval which is

$$t_{program} \leq T = \frac{1}{360 \text{ Hz}} = 2.78 \text{ msec.}$$

Since, at this point, the filter algorithm had not been developed, the following criteria were established concerning the software development.

1. Program in a serial fashion as much as possible. Conditional and unconditional branch or jump commands and subroutines calls, especially in looping operations, take unnecessary program time and should be avoided where possible.
2. Program in 8080A assembly language rather than in a high level language such as PL/M, PASCAL, or BASIC. Higher level languages, while easing the programming burden, include many "overhead" assembly language steps which do not necessarily enhance the program performance and can greatly decrease the program speed. Assembly language programming has the potential for efficient program code with a resulting faster filter program speed.
3. Overlap filter operations where possible. The only immediate visible opportunity to do this is to initiate the A/D function on the RTI-1200 from the filter program and perform other filter program functions during the 25 microseconds required to perform the conversion.

These criteria showed no concern for the amount of memory available in the microcomputer system. However, it was not felt that available memory space would be a problem when developing the filter program on the MDS because of the large amount of memory available on this particular system.

SYSTEM IMPLEMENTATION

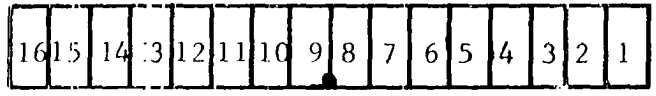
Number Representation

It was decided to use fixed point arithmetic in the filter system due to the ease of handling these types of numbers. Although a larger input dynamic range is possible with floating point arithmetic, the fact that the RTI-1200 provides and accepts fixed point data values as does the hardware multiplier, made the selection of fixed point arithmetic that much more obvious.

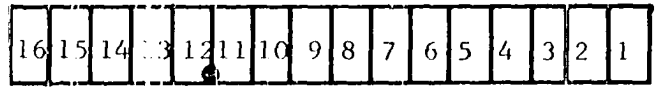
Since the hardware multiplier board will accept 16 bit input operands, it was decided to make the filter parameters or coefficients 16 bits long also.

Examination of the results of the PL/1 simulation program SCAN showed that the nonfraction portion of the digital filter parameters did not exceed a value of 5_D and all of these parameters contained a fractional portion. Therefore, a binary point would be assumed between the eighth and ninth bit positions of the 16 bit filter parameter and the most significant bit would be the sign bit. This convention is illustrated in Figure 13a.

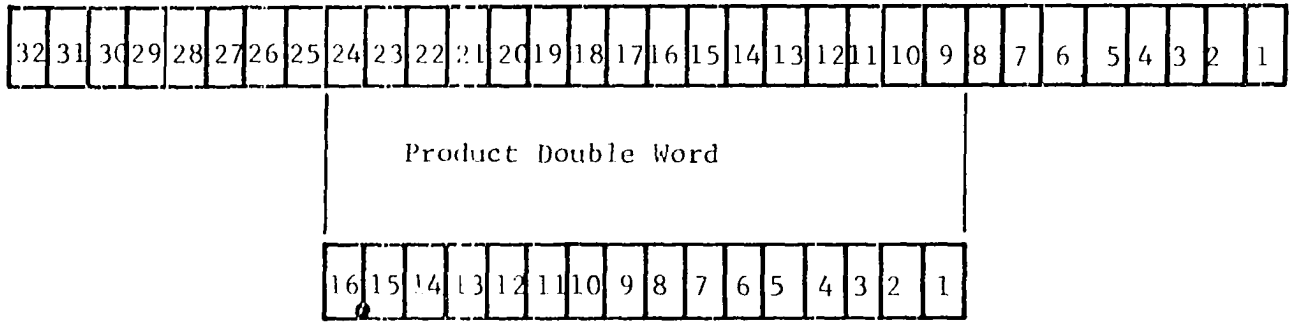
The manufacturers of the RTI-1200 had already set the fixed point representation format for both the input and output. As illustrated in Figure 13b, the lower order 12 bits represent the fixed point number magnitude while the high order 4 bits are sign bits.



a. Filter Parameter



b. RTI - 1200



c. Truncated Product Word

Figure 13. Filter number representation conventions

Since the product of two 16 bit operands is 32 bits long, some means of extracting 16 bits of the product bits and still maintaining the integrity of the product value was needed. Since the RTI-1200 input data form is 12 bits long and the largest possible filter parameter value would be 11 bits long, the useful product length would be 23 bits plus a sign bit long. If the fractional portion of the filter parameters or coefficients was truncated from the least significant portion of the product, only 16 bits of the useful product would remain. Therefore, a 16 bit product value would be the middle 16 bits of the 32 bit product possible from the hardware multiplier. This is illustrated in Figure 13c.

Data Overflow

If the hardware multiplier function was used to sum the products of the input operands, the possibility of data overflow exists. While the 35 bit accumulator in the TDC 1010J can handle the largest possible such sum that this adaptive filter system can generate, the 16 bits extracted from the product may not include all of the resulting high order product bits which results from the data overflow.

Examination of the filter parameter or coefficient values generated by the PL/1 simulation program SCAN reveals that the three center-frequency-dependent feedback parameter absolute values become larger as the filter center frequency moves in either direction from the center of the baseband. This would indicate that, if the input signal peak

amplitude remains constant throughout the baseband, data overflow is more likely to occur as the filter center frequency moves away from the center of the baseband.

This exact situation was demonstrated during the implementation of this digital filter system. As the filter configuration was changed to have a center frequency toward the extremes of the baseband, it was necessary to decrease the input signal amplitude to insure that data overflow and the resulting filter instability did not occur.

Two possible solutions to this problem were studied. Oppenheim and Schafer (11) discussed examining the output of the digital filter to detect an overflow condition. If this condition were detected, then the output was set to the highest possible output value or, in other words, saturated. This result would result in stable filter operation, but would also yield a distorted output. Also, in order to be fully effective for the direct canonic recursive configuration selected for this digital filter system, the sum of the products of the delayed inputs and filter feedback parameters or coefficients would need to be examined in order to fully protect the filter system from instability due to data overflow.

The second possible solution, discussed by Antoniou (1), was to scale the filter input signal so that the input signal amplitude to filter will always be less than some upper bound, that is,

$$x_i \leq M$$

where x_i is the digital filter input value and M is the upper bound of the

input values. This would insure that the input signal amplitude would never be too large to cause data overflow to occur.

Scaling of the input signal amplitude was selected as the approach to be implemented in this case. Monitoring the sum of the products values from the hardware multiplier board would be a more complex solution and would require more program time if implemented. Also, the resulting distorted output was an undesirable side effect. However, Antoniou's approach to scaling would set a single upper bound on the input signal amplitude. A trade-off would be required between what portion of the baseband would be used and how much the dynamic range of the inputs of the digital filters configured toward the center of the baseband would need to be compressed.

By determining the input scale factors for each digital filter configuration and using the appropriate scale factor with each filter configuration, the dynamic range of the possible digital filter configurations possible within the baseband will not be sacrificed and all such configurations would be practical.

Since the adaptive routine compares the digital filter peak output with a predetermined threshold value, different filter signal input amplitudes due to input scaling will result in different filter peak outputs for different filter configurations. Therefore, a unique threshold value will be needed for each different configuration.

Digital Filter Configuration

Since the hardware multiplier can sum the products of its input operands in its own accumulator, the choice of the direct canonic recursive configuration for the digital bandpass filter, shown in Figure 9, should now be apparent. When the hardware multiplier is set in this mode, the sum of the products of the delayed input values and the appropriate filter feedback parameters or coefficients can be calculated without changing the hardware multiplier mode. The product of the new digital input value and its scale factor (the scale factor is not shown in Figure 9) can also be added before changing the hardware multiplier mode to extract the sum of products value from the hardware multiplier accumulator. Finally, with the feedback plus scaled input value obtained and the hardware multiplier in the sum of products mode, the filter digital output value can be formed by summing the products of the appropriate delayed input values and the appropriate feedforward filter parameters or coefficients.

Note that in using the direct canonic recursive representation, only six hardware multiplier mode changes are required for each iteration of the digital filter routine. The digital filter system could have been configured in either the parallel or cascade canonic representations. However, the parallel canonic form shown in Figure 4 would have required a hardware multiplier mode change after each multiplication and the summation of the outputs of the parallel portions of the filter would have to be done by the 8080A which would require more program execution time.

If the filter were configured in the cascade canonic representation, the hardware multiplier mode would again require changing after each multiplication. These extra mode changes could also require more program time.

A filter bandwidth of 15 Hz was selected to demonstrate the operation of the filter system. While other bandwidth values could have been selected for the filter there is a practical minimum bandwidth limit. This limit is discussed in a later section.

Since the entire filter program plus the adaptive program must be complete within 2.78 milliseconds after the occurence of the R-C "PACER" generated interrupt, the fastest digital filter routine should be used. It is obvious that the direct canonic recursive representation yields the fastest digital filter program.

Adaptive Routine

The adaptive program will perform a binary tree search of the baseband using predetermined fixed center frequencies. A run of the PL/1 simulation program SCAN was made showing all of the bandpass digital filter parameters for a filter with a 15 Hz bandwidth and for all possible even integer valued center frequencies within the baseband. This yielded a list of 83 center frequency values ranging from 8 Hz to 172 Hz in steps of 2 Hz. A binary tree was constructed from this list of center frequencies and is shown in Figure 14. Note that this binary tree only contains 49 nodes or frequencies. This is partially due to the

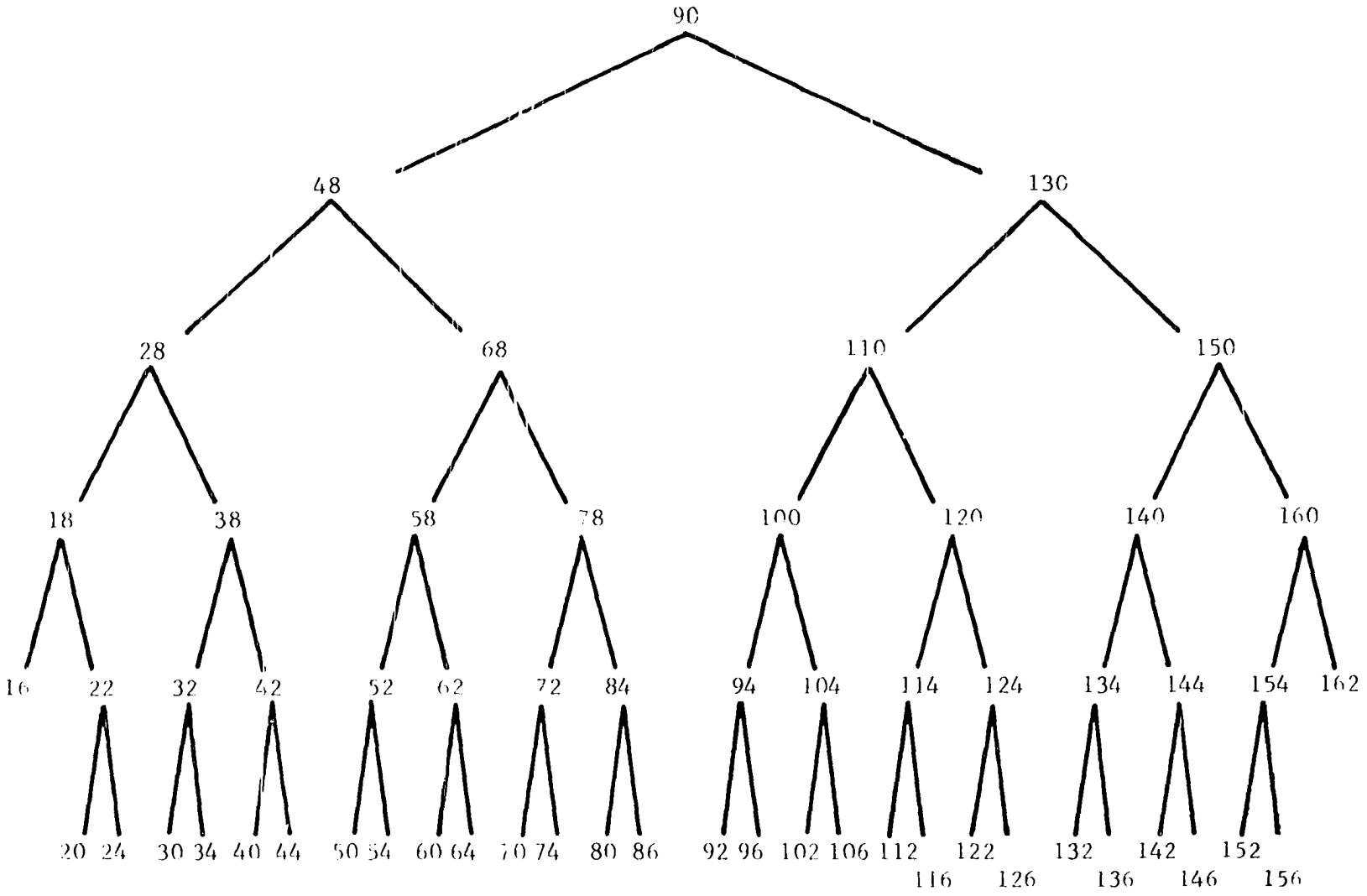


Figure 14. Adaptive routine baseband binary tree

way the tree was constructed and also because, for some of the extreme high and low filter configuration center frequencies, there was no input scale factor greater than zero which would yield a stable filter configuration. For this reason, some of the frequencies on the extreme edges of the baseband are omitted from the tree. However, each tree node has two subnodes or none at all.

A look-up table called BRANCH was constructed from this binary tree. Each frequency value or tree node on the tree was encoded with an 8 bit byte. This 8 bit byte serves as a pointer to a 16 bit word in the look-up table BRANCH. The low order 8 bits of this 16 bit word contains the pointer to the right subnode of this node while the high order 8 bits contains the pointer to the left subnode. Those nodes which have no subnodes point to a terminating symbol $FFFF_H$.

The look-up table BRANCH is located in a contiguous 256 byte block of memory. An entry in BRANCH can be found by loading the high order 8 bit address byte for the first entry in BRANCH into the 8080A H register and the 8 bit node pointer into the L register which then forms the address for a right subnode.

The same pointer values from BRANCH are used to access appropriate values in the five other look-up tables. Tables VAFT1, VAFT2, and VAFT3 contain the 16 bit filter parameter or coefficient values for FACTOR1, FACTOR2, and FACTOR3, respectively, to allow the digital filter system to be configured to any of the center frequencies presented in the binary tree.

The look-up table SCALER contains the appropriate 16 bit input scale factors for each digital filter configuration presented in the binary tree. These values were determined experimentally and are all based on a peak-to-peak input amplitude limit of 17 volts.

The look-up table THRESH contains all of the appropriate threshold comparison values for each of the digital filter configurations shown on the binary tree which has a nonterminating subnode. These values were also determined experimentally and are also based on a peak-to-peak input amplitude of 17 volts.

The same pointer values from BRANCH are used to access the appropriate values in the five other look-up tables. The 8 bit pointer is used as a pointer to a table value while the appropriate 8 bit high order byte is used to point to the contiguous 256 byte block of memory containing the appropriate table.

When implementing the adaptive digital filter system, it was sometimes found that the filter would become unstable after the feedback filter parameters or coefficients FACTOR1, FACTOR2, and FACTOR3 were changed. This instability was observed with an oscilloscope at the filter system output as an unordered or random-appearing display of output voltage levels. This was especially true when the digital filter was configured to have a center frequency toward the upper and lower limits of the baseband. The reason for this occurrence is that sometimes the set of delayed input values from the previous set of filter parameters is too large when compared to that set of delayed input values which

might be encountered when the filter is configured with the succeeding set of parameters. When the old set of delayed input values is multiplied by the new set of filter parameters and the sum of the feedback products formed, data overflow can and does occur.

It was discovered that this tendency toward instability could be removed if the delayed input values were reset to zero after each change in the center frequency dependent filter parameters, FACTOR1, FACTOR2, and FACTOR3. Proper prescaling of the input to the digital filter system also greatly reduces this tendency, especially when the filter center frequency was set at frequencies toward the extreme edges of the baseband.

Hardware Implementation

A block diagram of the digital filter system hardware integrated with the MDS system is shown in Figure 15.

Only one of the sixteen available analog inputs on the RTI-1200 is used as the input for the filter. Also, only one of the two digital to analog converters is used to monitor the filter output. The "PACER" R-C clock is preset to issue an interrupt to the 8080A microprocessor in the MDS at the sampling rate of 360 Hz. The filter system program is initiated with each occurence of this interrupt. The interrupt controllers in the MDS and on the RTI-1200 are reset by the filter system program at the conclusion of each iteration of the filter system program.

To accommodate the RTI-1200 and hardware multiplier printed circuit boards in the MDS it was necessary to remove one of the memory printed

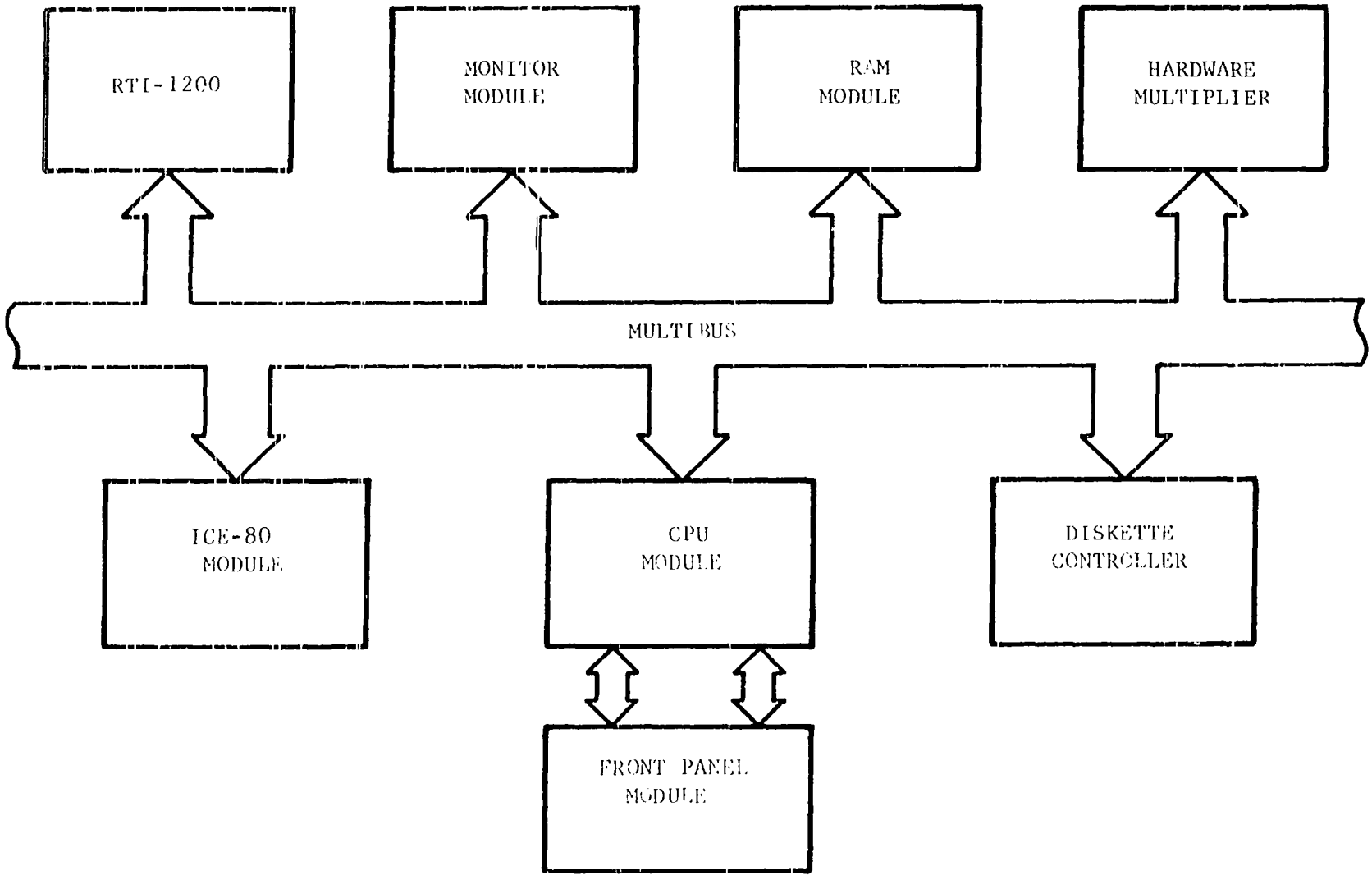


Figure 15. Digital filter system integrated with Intel Intellec 800 MDS

circuit boards from MDS. Since the MDS contained 64 K bytes of RAM, the removal of one of the memory boards left 48 K bytes of memory which is more than adequate to perform the function of this adaptive filter system.

Software Implementation

The adaptive filter system program was developed and programmed into nine separate program modules on the MDS. These nine program modules, listings of which are presented in appendix 3, are:

1. RTI-1200 address location module (INIT.AD);
2. Hardware multiplier address location module (INIT.MUL):
3. Digital filter parameter module (INIT.V4A);
4. Filter system variable module (INIT.VA2);
5. Filter parameter look-up table module (TABVAL.V2);
6. Filter system initialization routine module (INIT.V8);
7. Filter parameter initialization routine module (INIT.RST);
8. Bandpass filter routine module (BPF4TH.V5); and
9. Adaptive routine module (ADAP.V6).

These modules were all linked together on the MDS using the ISIS-II "link" command and the entire adaptive filter system program module was located in the MDS memory using the ISIS-II "locate" command.

The RTI-1200 and hardware multiplier address location modules define the memory mapped locations of the data and control signals for these hardware subsystems.

The digital filter parameter module defines the address locations of the seven bandpass filter parameters or multiplier coefficients. This module also contains the filter parameters to initialize the bandpass filter center frequency to 90 Hz which is the initial node of the binary tree. The filter parameter initialization routine loads these initial parameter values into the filter parameter locations.

The filter system variable module defines the address locations for the digital filter delayed input values as well as the other variables used by the bandpass filter and the adaptive routines.

The filter parameter look-up table module contains six data tables, BRANCH, VAFT1, VAFT2, VAFT3, SCALER, and THRESH, used by the adaptive routine.

The filter system initialization routine module contains the program which initializes all of the adaptive filter system values, parameters, and controls.

The filter parameter initialization routine module contains the program that initializes the bandpass filter center frequency to 90 Hz. This routine is called by an MDS system interrupt which can be activated automatically by the filter system initialization routine or manually from a switch on the front panel of the MDS cabinet.

The bandpass filter routine module contains the bandpass filter program. This program is the basis for the whole filter system and is called by an MDS system interrupt generated by the RTI-1200 R-C "PACER" clock which is set to the filter sampling rate.

The adaptive routine module contains the adaptive routine program. This program monitors the output of the bandpass filter routine and reconfigures the bandpass filter to determine a match between the bandpass filter center frequency and the input signal frequency.

SYSTEM OPERATION

The filter system initialization routine (INIT.V8) is the first program run when the adaptive filter system is activated. This program, first, initializes the MDS system interrupts to respond to an interrupt generated by the RTI-1200 R-C "PACER" clock and to an interrupt which calls the filter parameter initialization routine (INIT.RST). The system stack pointer is then initialized and the bandpass digital filter parameters or coefficients are initialized by the generation of a software interrupt which, in turn, calls the filter parameter initialization routine.

Next, the filter system initialization routine initializes the RTI-1200 functions. Specifically, the D-A converter is initialized to zero volts, the analog input gain is set to unity, the analog input channel is selected, and the R-C "PACER" clock system is activated.

The adaptive routine variables are then initialized. The peak output value is set to zero and the high order 8 bit pointers to the look-up tables are set.

At this point, the remaining digital filter parameters are initialized. The delayed input values are set to zero and the input scale factor is initialized.

Finally, the hardware multiplier board functions are initialized so that operand values can be input to the board.

The adaptive filter system initialization is now complete and the filter system now waits for the occurence of the first interrupt generated by the R-C "PACER" clock.

When the awaited interrupt arrives, program control is transferred to the bandpass filter routine (BPF4TH.V5). This program immediately initiates the sampling and digital conversion of the filter analog input since the RTI-1200 requires 25 microseconds to accomplish this function.

The memory location which controls the initiation of the multiplication function of the hardware multiplier board is then loaded into the 8080A B-C register pair. This step uses of the store accumulator indirect (STAX B) command, which requires only 7 machine cycles, to initiate a multiplication rather than a store accumulator direct (STA LDMULT) command which requires 13 machine cycles. This is a difference of 6 machine cycles per instruction or, since 7 multiplication operations are required for each iteration of the digital filter program, 42 machine cycles per program iteration. When the 10 machine cycles to store the multiplication initiation command location in the B-C register pair (LXI LDMULT) are considered, the overall time savings are 32 machine cycles or 16 microseconds of program time per iteration.

Now the filter operation actually begins with the formation of the sum of the products of the delayed input values and the feedback filter parameters or coefficients. When this summation is complete, the new sampled input value is in digital form. This new value is multiplied by the appropriate scale input value and the resulting product is added to the sum of the feedback values.

The filter output value is now formed by summing the product of the new feedback value and FACTOR5 and the products of the appropriate delayed input values and the appropriate filter feedforward parameters

or coefficients. This sum is then submitted to D-A converter to form the analog output.

During this process, the filter delayed input values have been shifted appropriately in preparation for the next iteration of the bandpass filter routine.

Except for the reset of the hardware multiplier to input new operands and resetting the RTI-1200 and MDS interrupts, the remainder of this routine is really a part of the adaptive routine. The filter peak output is detected and saved for use by the adaptive routine (ADAP.V6). Also, a check of the position of the manual switch on the hardware multiplier board is made to determine if the adaptive routine is to be exercised. If the adaptive function is to be performed, control is transferred to the adaptive routine. Otherwise, the filter system goes into a standby mode to await the next occurence of the interrupt generated by the R-C ""PACER" clock on the RTI-1200.

The adaptive routine (ADAP.V6), though appearing to be a very long program, is actually run in three separate segments. A flow chart diagram of the adaptive routine is shown in Figure 16.

In the first segment of the adaptive algorithm, the peak output value of the filter configured with its center frequency at the node frequency is compared with the corresponding threshold value. If the peak output value does exceed or equal the appropriate threshold value, the adaptive routine assumes that the center frequency of the digital bandpass filter or node frequency is the same as the filter input signal

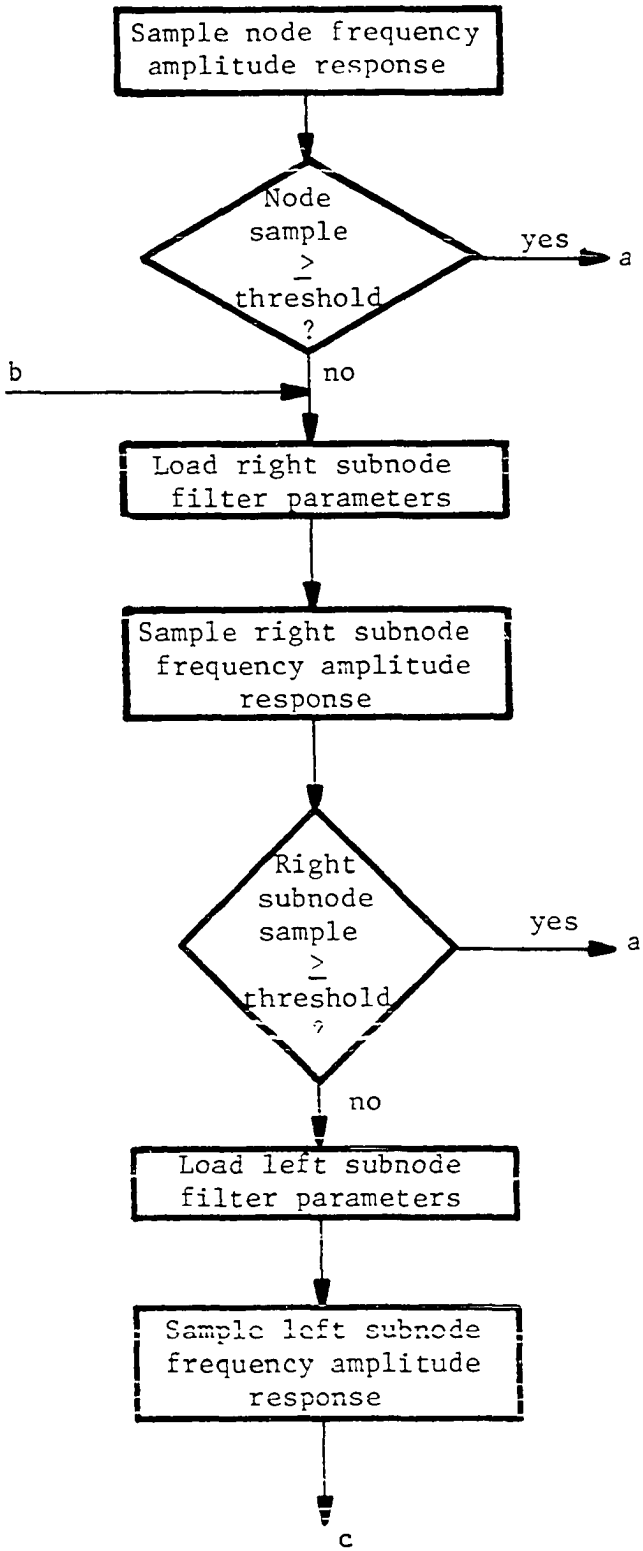


Figure 16. Adaptive routine flow chart

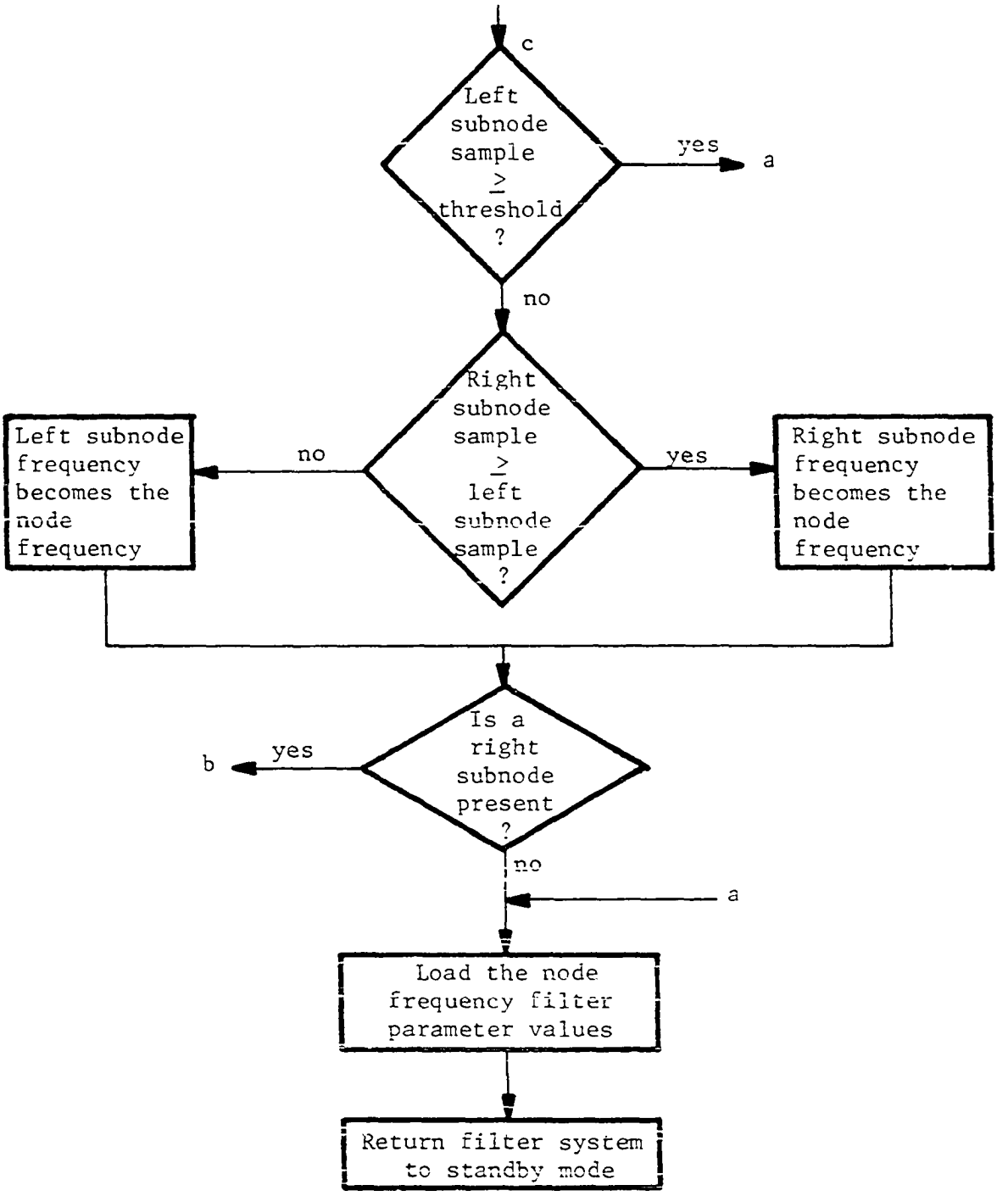


Figure 16. Continued

frequency and any further adaptive routine operations are suspended. Otherwise, the adaptive routine checks the BRANCH look-up table for a right subnode frequency. If there is no right subnode frequency, the adaptive routine assumes that the node frequency or filter center frequency matches or closely matches the input signal frequency and further adaptive routine operations are suspended. Recall that the binary tree is balanced so that if a right subnode is present a left subnode will also be present, and conversely.

If there is a right subnode frequency to be examined, the adaptive routine loads the appropriate filter parameter values from look-up tables VAFT1, VAFT2, and VAFT3, respectively, into filter parameter locations FACTOR1, FACTOR2, and FACTOR3 which configures a digital bandpass filter with the right subnode center frequency. The appropriate scale factor value is loaded into the SCALE location from the SCALER look-up table for use by the digital filter routine, the digital filter delayed input values are reset to zero, and a pointer is set to locate the next part of the adaptive routine to be performed when control is returned to that routine.

The last step of this portion of the routine resets the RTI-1200 and MDS system interrupts and returns the filter system to the standby mode where, once again, the system awaits the next occurrence of the R-C "PACER" clock interrupt.

For the next 24 occurrences of this interrupt, only the bandpass filter routine will be run. The first 8 occurrences allow the filter to adjust to the change in filter parameters. During the last 16 iterations

of the digital filter program, the output of the digital filter is monitored to determine the filter peak output value. After the twenty-fourth iteration of the digital filter routine has completed, the adaptive filter routine is again called. As in the first segment of this routine, the filter peak output of this right subnode center frequency filter configuration is compared with the appropriate threshold value to determine if the center frequency (right subnode frequency) of the digital filter matches the input signal frequency. This portion of the routine operation is the same as in the first segment. Next, the digital filter is reconfigured to examine the left node frequency. This reconfiguration procedure is the same as was previously described. Also, a new factor is loaded into SCALE, the delayed input values reset to zero, the new routine return pointer is set, the RTI-1200 and MDS system interrupts reset, and the filter system is again returned to the standby mode.

Again, only the bandpass digital filter routine is activiated with the next 24 occurrences of the R-C "PACER" clock generated interrupts. As before, the first 8 iterations of the digital filter routine allow the filter to adjust to the change in filter parameters and the filter peak output value is determined during the remaining 16 iterations.

At the completion of this series of digital filter routine iterations, the adaptive routine is again called and the filter peak filter output value for the left subnode frequency is compared with the appropriate threshold value. If the filter peak output value does not exceed the threshold value for the left subnode center frequency digital filter

configuration, the peak output value is compared with the peak output value for the right subnode center frequency digital filter configuration. The subnode filter configuration which yields the larger filter peak output value now becomes the new node frequency. The digital filter is then reconfigured to have a center frequency at the new node frequency. A pointer is then set to the beginning of the first segment of the adaptive routine so that the adaptive process may begin again after the 24 iterations of the digital filter routine.

In order to be able to study the performance of the adaptive routine, this adaptive filter routine does not include the capability to track the frequency on an input signal. The filter system could be modified to do so by continuing to monitor the peak filter output value. If this value ever dropped below its appropriate threshold value, the system would be reset to have a node frequency of 90 Hz and repeat the iterations of adaptive routine.

SYSTEM EVALUATION

This adaptive filter system will be evaluated by examining the performance of the basic digital bandpass filter and the performance of the adaptive routine with the basic filter routine.

Bandpass Digital Filter Performance

The PL/1 program RESPONS was written to present the frequency response for the digital bandpass filter. This program accepts the upper and lower cutoff frequencies and the upper bound of the filter baseband and calculates the filter frequency response for the digital bandpass filter within the baseband in 5 Hz increments. The program performs this function by calculating the locations of the digital filter poles and zeros on the z-plane and then geometrically determining the distances between the filter poles and zeros and the point $e^{j\omega T}$ where

ω = any frequency in radians/second within the baseband; and

T = the sampling interval which is 2.78 milliseconds in this case.

The overall frequency response of the digital filter is the quotient of the product of the distances between $e^{j\omega T}$ and the filter system zeros divided by the product of the distances between $e^{j\omega T}$ and the filter system poles.

The frequency response of the actual digital filter was measured by applying a fixed amplitude sine wave to the digital filter input. The input sine wave frequency was varied and the digital filter peak output amplitude was measured using an oscilloscope.

The results of this program are compared with the measured frequency response of the implemented digital bandpass filter. This comparison is shown in Figure 17.

Figure 18 shows the time responses of the bandpass digital filter configured with a 90 Hz center frequency when constant amplitude sine waves with frequencies of 83 Hz, 90 Hz, and 98 Hz are applied to the filter input. Note that 83 Hz and 98 Hz are the bandpass filter 3 dB bandwidth frequencies. The line segments shown in the representations in Figure 18 are the digital filter outputs. Each line segment is 2.78 milliseconds long or the duration of the sampling interval.

If the input sine wave is simultaneously displayed with the digital filter output on an oscilloscope which is triggered by the input signal, the leading edges of the output filter line segments trace the output sine wave response. Note that unless the input signal has a harmonic of 360 Hz, the filter system sampling rate, the filter output line segments will be moving to form such a sine wave trace.

Adaptive Routine Performance

The adaptive routine was tested by applying a 17 volt peak-to-peak sine wave signal at different frequencies to the filter system input, activating the adaptive routine, and noting the node frequency on which the adaptive routine settles. Repeated reference to Figure 14 will be helpful when studying the following evaluation summary.

The first frequency tested was the initial node frequency, 90 Hz. With the input signal amplitude set to the proper value, the adaptive

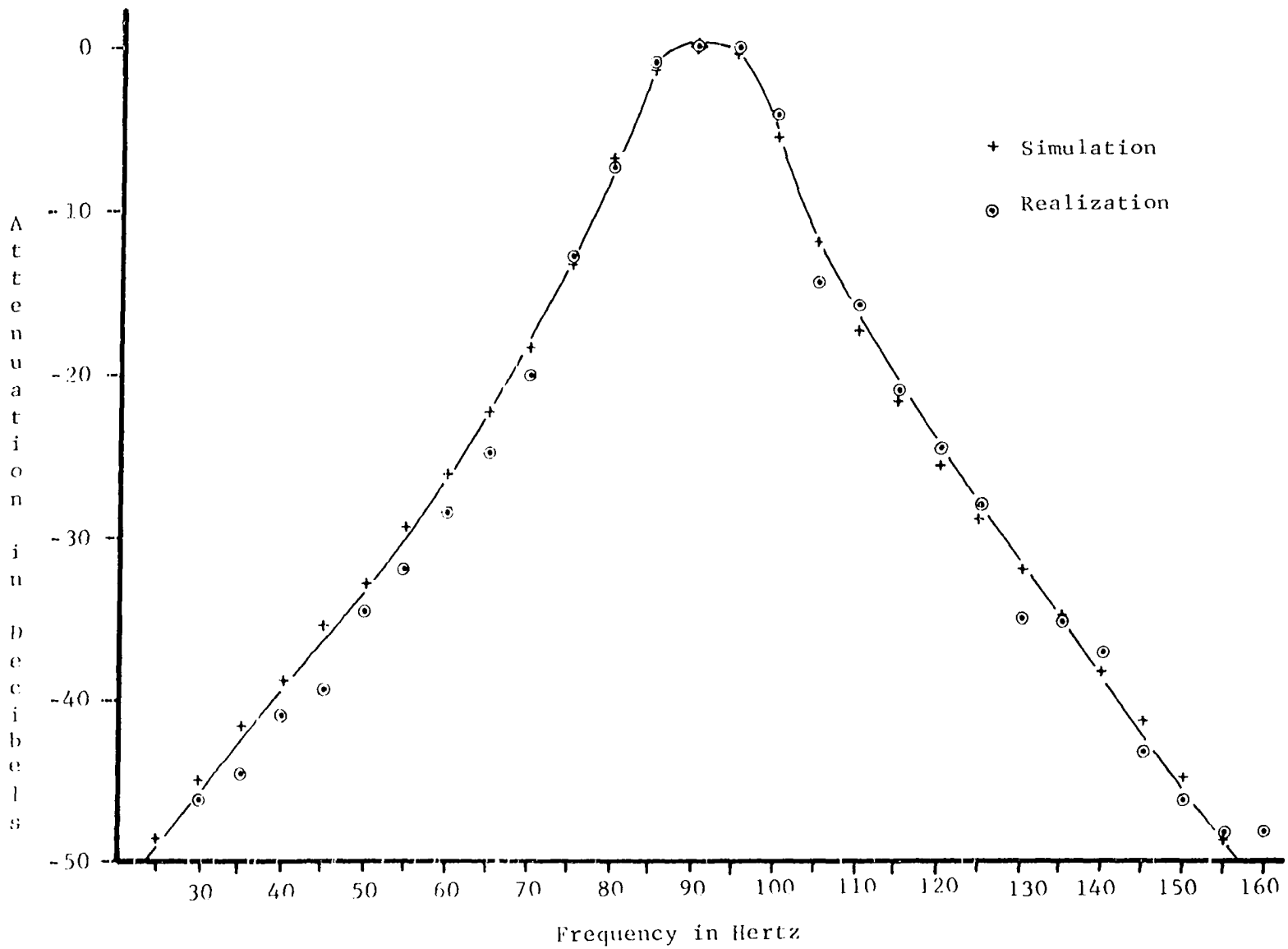
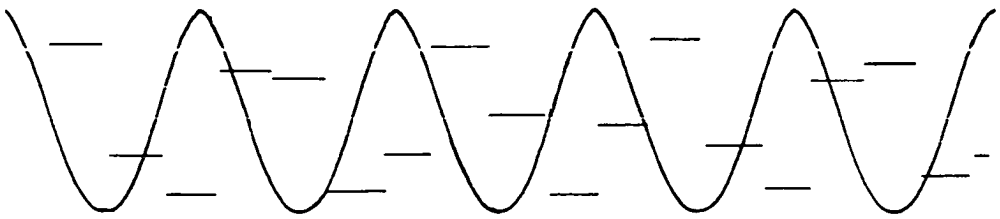
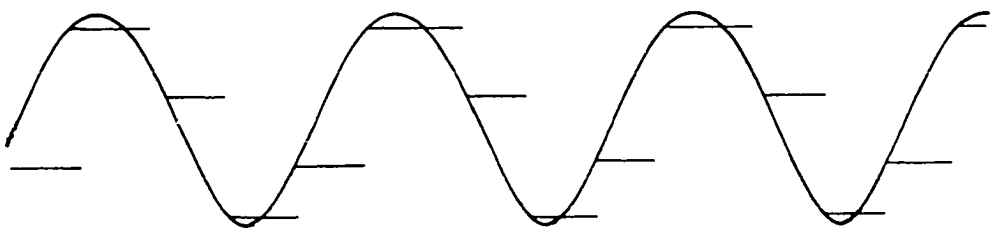


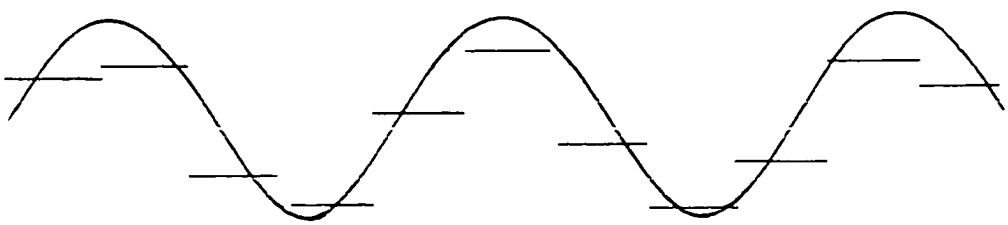
Figure 17. Digital Filter frequency response



98 Hertz



90 Hertz



83 Hertz

Figure 18. Digital filter time response

routine would maintain the filter system center frequency at 90 Hz. However, if the input amplitudes were too low, the filter peak output value would not equal or exceed the predetermined threshold value in the look-up table THRESH and the adaptive routine would commence its search down the binary tree and yield an erroneous filter center frequency.

The next input signal frequencies examined were the subnode frequencies of the initial node frequency, 48 Hz and 130 Hz. In both of these cases the adpative routine caused the digital filter center frequency to settle onto these respective subnode frequencies. Small modifications to the appropriate threshold values in the look-up table THRESH were necessary to obtain proper operation of the adaptive routine.

The successive subnode frequencies were examined in the same manner and similar results were obtained until the fourth tier of nodes below the initial node was reached. At this point, a few center frequency adaptation errors occurred. As mentioned in the preceding paragraph, in some cases slight modifications to the appropriate threshold values were required to assure proper operation of the adaptive routine. In a few cases it was necessary to modify the scale factor values in the look-up table SCALER to reduce the systems tendency toward instability.

Where erroneous performance became noticeable was in the fourth and last tiers of node frequencies. In most cases, the adaptive routine would settle the filter center frequency on the frequency of a parent node in the fourth tier of nodes. Adjustment of the threshold values of the fourth tier node frequencies could not correct the situation.

The reason for the above noted incorrect operation is that the frequencies in the last tier of nodes are within the 15 Hz filter passband of the filter configuration of one of the nodes in the upper tiers of the binary tree. Because of the flat passband response of the Butterworth filter, the filter will yield almost the same peak output values for all of the frequencies in the filter passband. It is therefore almost impossible to find a threshold value which will differentiate between the peak output values resulting from an input signal at the center frequency of a bandpass filter configuration and other input signals within the passband of that filter configuration.

The entire adaptive digital filter system requires 1912 bytes of memory storage. Of this 1912 bytes, 966 bytes are required for the six look-up tables, 888 bytes are required for the filter system program software, and 58 bytes are required by the filter system program software for temporary storage.

The processing time required for the adaptive digital bandpass filter system was calculated and examined. Because the adaptive routine portions are accessed only a few times before the bottom of the binary tree is reached, it was difficult to measure these times experimentally since the accessing of these segments of code is not periodic. For this reason the processing times were calculated rather than measured.

These times were calculated by summing the clock states for all of the instructions in each applicable portion of each routine. The clock state sums were then multiplied by the 0.5 millisecond 8080A clock period to obtain the processing time for each applicable program segment. It

should be noted that the clock period of the 8080A in the MDS was measured on an oscilloscope to be 0.5 milliseconds.

The initialization routine processing time was not calculated since that routine is run only once and the duration of that routine does not affect the performance of the rest of the filter system.

The digital bandpass filter which was implemented by itself requires 0.5235 milliseconds to complete its function without jumping to the adaptive routine. This time is considerably less than the filter system sampling interval of 2.78 milliseconds. If the filter system sampling interval was decreased to 0.5235 milliseconds, a sampling rate of 1910 samples per second could be realized. This means that a nonadaptive fourth order digital bandpass filter could be implemented with a baseband of up to 955 Hz.

If the adaptive routine was added to the digital bandpass filter operation, a worst case processing time of 0.982 milliseconds is required. This value is also well below the 2.78 millisecond sampling interval used in this filter system. If the sampling interval were set to 0.982 milliseconds, a sampling rate of 1018 samples per second is possible which results in a maximum possible baseband of 509 Hz.

System Features

The adaptive digital bandpass filter has several advantages to recommend it. The first is, assuming that the input scale factors are correct, the stability of the digital filter is assured provided that the input signal amplitude remains below the preset upper limit.

The adaptive filter system program can also be used with different sizes of filter basebands by providing the proper data in the look-up tables and adjusting the sampling rate, that is, the R-C "PACER" clock frequency on the RTI-1200. No change in the adaptive filter system program is required.

Other classes of filters, such as Chebyshev or elliptic, can also be implemented using this adaptive filter system program. Again, only modification of the data in the look-up tables is all that is required to make this change.

The resolution of the adaptive search can be modified, up to a point, by also changing the data in the look-up tables. The BRANCH look-up table can accommodate up to 128 nodes. A different approach to locating entries in the BRANCH look-up table could be used to permit utilization of all 128 possible nodes. If more than 128 nodes were required, the adaptive routine would need to be modified to accommodate a larger BRANCH look-up table.

System Deficiencies

The adaptive digital bandpass filter does suffer from several deficiencies which degrade the performance of the system.

One notable deficiency is that the threshold values contained in the THRESH look-up table are all based on a fixed input signal amplitude, in this case 17 volts peak-to-peak. However, if the input signal amplitude rises above or drops below that fixed value, the values in the

THRESH look-up table are no longer valid and incorrect operation of the adaptive routine is practically assured.

When an input signal is close in frequency to the center frequency of the digital bandpass filter, but not matching, the adaptive routine will sometimes incorrectly indicate a match. The basic cause of this incorrect operation is the flat passband response of the Butterworth filter. If an input signal frequency falls within the passband of a particular filter configuration, but does not match the center frequency of the center of that filter configuration, the filter response to that input signal could be a peak output value which may exceed the appropriate threshold value for that particular configuration. This would in turn cause an incorrect node selection to be made by the adaptive routine with the end result being that an exact match of the input signal frequency and the digital bandpass filter center frequency would be impossible.

Three possible remedies to this deficiency are apparent. First, an exact match between the input signal frequency and the filter input frequency may not be necessary, depending on the particular application. In this case, nothing needs to be done.

Another solution might be to use a filter type which has a not-flat passband, such as a Chebyshev or elliptic filter. With this type of filter some attenuation is possible within the passband which may help to make the appropriate threshold value more unique in determining a match of the digital bandpass filter center frequency with the input signal frequency.

A third solution might be to configure the digital bandpass filter with a narrower passband. There is a bandwidth limit, however, where values for filter parameters FACTOR5 and FACTOR7 round off to 0000_H. This appears to occur when the filter bandwidth reaches 8 Hz. It is not known what effect this truncation of the filter parameters might have, but it is possible that incorrect filter operation could result.

Possible Future Work

This adaptive digital bandpass filter system, while having been demonstrated to be a workable system, lacks many refinements which would certainly enhance its overall operation.

An obvious improvement would be to implement this system using a faster 8 bit microprocessor and possibly one with a more efficient instruction set. If this filter system were implemented on a Zilog Z-80A based microcomputer system with no changes to the software, this same filter system could operate at twice the program speed of the demonstrated 8080A based filter system configuration. This would mean that the filter baseband could be twice the size possible with the 8080A based system.

In addition, the Z-80A contains more registers and its instruction set contains more addressing modes than the 8080A so more efficient coding of the entire filter system software would be possible.

Greater program efficiency, faster program operation, and the elimination of the hardware multiplier are all possible improvements to the filter system if it were implemented on the new generation of 16 bit microprocessors such as the Intel 8086, the Zilog Z-8000, or the Motorola 68000.

While all of these devices have multiplication instructions in their repertoires, the TRW TDC1010J hardware multiplier device still multiplies at a faster speed. However, these 16 bit microprocessors do not require the extensive overhead programming steps to perform a multiplication which are necessary when multiplying with the hardware multiplier. As a result, the overall time required to perform a multiplication using a 16 bit microprocessor is less than that required to perform the same task using the hardware multiplier.

In addition, all three of these 16 bit microprocessors contain more effective addressing modes in the instruction sets than those in the 8080A instruction set. The 16 bit microprocessors also are designed to operate on 16 bit data more efficiently than the 8080A. Consequently, a faster, more efficient adaptive digital filter system software implementation is possible if a 16 bit microprocessor based system is used. Such a faster routine might be developed.

The present adaptive digital bandpass filter system is very tightly constrained by the lack of range of the threshold values contained in the THRESH look-up table. The development of an algorithm to dynamically alter these threshold values as the filter input signal amplitude changes would greatly enhance the dynamic range of the input of the filter system.

In like manner, an algorithm which would dynamically modify the input scale factors as the peak input amplitude changed would also enhance the filter system dynamic range. Such an algorithm would optimize

the input signal amplitudes to the digital bandpass filter so that these amplitudes would be as large as possible and yet not so large that the filter configurations are driven to instability. This algorithm could also be linked with an algorithm similar to the one previously described to optimize both the input scale factors and the threshold values.

A simple algorithm to make the demonstrated adaptive digital filter system truly track an input signal was described in an earlier section. This algorithm required that the adaptive routine perform the complete adaptive algorithm if the decreasing peak filter output amplitude indicated a shift in input signal frequency. An algorithm could be developed which, when having sensed a shift in input signal frequency, could search back up the binary tree structure to find a match of the filter system center frequency and the input signal frequency. This approach would be a less radical and abrupt change of the digital parameters and would provide for a smoother transition between filter configurations.

Such an algorithm would require a more complex scheme to keep track of parent nodes so that the binary tree could be traversed from bottom to top as well as from top to bottom.

In its present form, the adaptive digital bandpass filter system tries to match its center frequency with the first input signal it encounters that causes the filter output to equal or exceed the appropriate threshold value. A useful algorithm could be developed that seeks to match the filter system center frequency with the input signal which has the largest amplitude.

The present filter system operates only on continuous input signals. Recall that the original motivation for developing this adaptive digital filter system was to monitor heart rate information. However, the heart signal is not a continuous signal. Therefore, a useful algorithm would be one which still matches the digital filter center frequency to the input signal frequency, but where the input signal may no longer be continuous.

It was noted in an earlier section that the adaptive routine will sometimes provide an incorrect match of the filter system center frequency to the input signal frequency due to the flatness of the Butterworth filter passband. It was also noted that a narrower filter passband could help correct this deficiency. There could, however, be instances where a precise filter center frequency-input signal frequency match is desired as well as a wider filter bandwidth. Figure 19 shows the block diagram of a possible implementation of such a filter. In this configuration the narrow bandwidth version of the adaptive filter system would serve as an input center frequency tracker. The tracker would use the pointers it uses in its adaptive routine to locate applicable filter parameters in other look-up tables to modify the slave filter.

Note that the slave filter could be almost any type of filter with almost any size bandwidth, even a bandstop filter. Also, there could be multiple slave filters all adapted by the tracker.

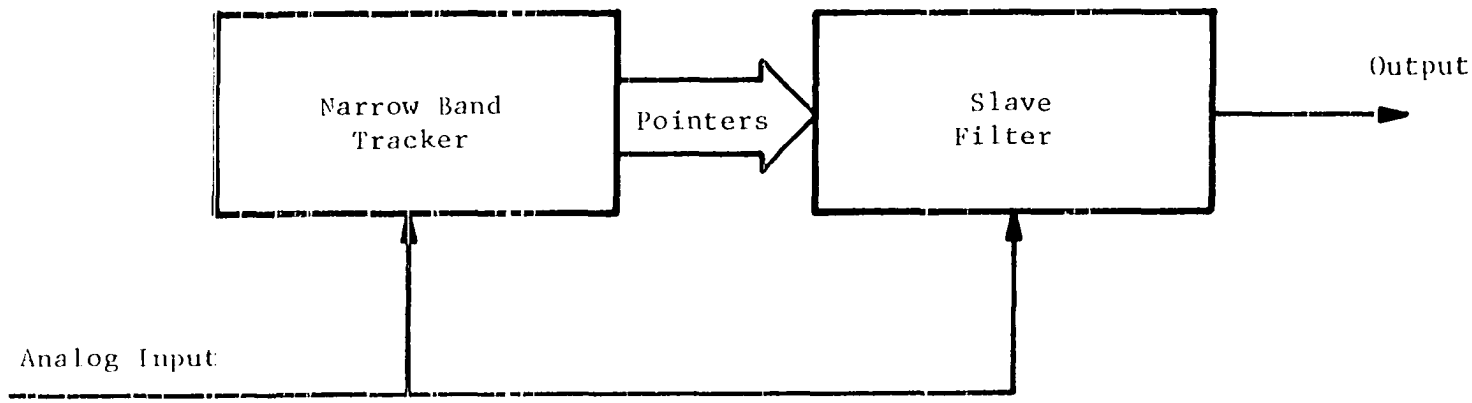


Figure 19. Tracker-slave digital filter

CONCLUSIONS

It has been demonstrated that a general purpose microprocessor, in this case the Intel 8080A, can be used to implement a real-time digital filter system if an external hardware multiplier is included in the system hardware implementation. The basic fourth order digital bandpass filter developed for this dissertation, not to mention the adaptive routine, would not have been possible without the inclusion of the TRW TDC 1010J hardware multiplier.

A faster 8 bit general purpose microprocessor, such as the Zilog Z-80A, will enhance the performance of a filter such as the one developed for this dissertation simply because of its faster processing speed. A microprocessor like the Z-80A can further enhance the performance such a filter system by making use of the more powerful addressing modes available on that microprocessor.

Despite the faster speed of the Z-80A, a fourth order bandpass digital filter implemented on the Z-80A based system should include an external hardware multiplier.

The use of an 8080A based microcomputer system implemented as a fourth order non-adaptive digital bandpass filter with a sampling rate of 360 samples per second, but without an external hardware multiplier, is unrealistic. The time required to perform the software multiplications is the greatest constraint. If the sampling rate were smaller, yielding a longer sampling interval, a fourth order bandpass filter could be

implemented. However, a lower sampling rate results in a smaller baseband or frequency range of interest which in turn limits the usefulness of the digital filter.

An adaptive version of the above mentioned filter is again unrealistic due the time required to perform both the filter and adaptive functions. As before, the best hope of obtaining a realizable adaptive filter system is to lower the sampling rate which, again, limits the usefulness of the filter.

The latest generation of 16 bit microprocessors shows great potential for implementation of digital filter systems. These devices are, generally, at least as fast as the earlier 8 bit devices. The three prominent 16 bit microprocessors available today, the Intel 8086, the Zilog Z-8000, and the Motorola 68000, all include a 16 bit by 16 bit multiplication operation in their instruction sets. These multiplication instructions require fewer overhead operations than the TRW TDC 1010J based and would result in more concise and efficient coding for multiplication.

The use of any of these three devices in an adaptive digital bandpass filter system similar to the one described in this dissertation will result in a simpler hardware configuration (since the external hardware multiplier is no longer necessary), a more efficient software package, and a digital filter system with greater performance than is possible with a general purpose 8 bit microprocessor based system.

BIBLIOGRAPHY

1. Antoniou, Andreas. Digital Filters: Analysis and Design. New York: McGraw-Hill Book Company, 1979.

2. Childers, Donald; and Durling, Allen. Digital Filtering and Signal Processing. St. Paul: West Publishing Company, 1975.

3. Constantinides, A. G. "Synthesis of Digital Filters from Continuous Filter Data." In Introduction to Digital Filtering, pp. 47-59. Edited by R. E. Bogner and A. G. Constantinides. New York: John Wiley and Sons, 1975.

4. Feintuch, P. L. "An Adaptive Recursive LMS Filter." Proceedings of the IEEE 64, No. 11 (November 1976):1622-1624.

5. Holsinger, William P.; Kempner, Kenneth M.; and Miller, Martin H. "A QRS Preprocessor Based on Digital Differentiation." IEEE Transactions on Bio-Medical Engineering BME-18, No. 3 (May 1971):212-217.

6. Intel Corporation, ISIS-II System User's Guide No. 98-306B. Intel Corporation, Santa Clara, Calif., 1975.

7. Intel Corporation. Intel Multibus Interfacing. Application Note AP-28A. Intel Corporation, Santa Clara, Calif., 1979.

8. Intel Corporation. 8080 Microcomputer Systems User's Manual. Intel Corporation, Santa Clara, Calif., 1975.

9. Jenkins, W. K. "Architectures for Microprocessor-Based Adaptive Digital Filters." Proceedings of the Twenty-first Midwest Symposium on Circuits and Systems 21 (1978):148-152.

10. Johnson, C. Richard, Jr.; Larimore, Michael G.; Feintuch, Paul L.; and Bershad, Neil. "Comments on and Additions to 'An Adaptive Recursive LMS Filter'." Proceedings of the IEEE 65, No. 9 (September 1977):1399-1402.

11. Oppenheim, Alan V.; and Schafer, Ronald W. Digital Signal Processing. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1975.

12. Parikh, D.; and Ahmed, N. "Adaptive Algorithm Considerations of IIR Filters." Proceedings of the Twenty-first Midwest Symposium on Circuits and Systems 21 (1978):369-373.

13. Parikh, D.; Ahmed, N.; Johnson, C. Richards, Jr.; and Larimore, M. G. "On 'An Adaptive Algorithm for the IIR Filters'." Proceedings of the IEEE 66, No. 5 (May 1978):585-588.

14. Soderstrand, Michael A. "Applications of Microprocessors in Digital Signal Processing." Proceedings of the Twenty-first Midwest Symposium on Circuits and Systems 21 (1978):153-157.

15. Wartak, Josef. Computers in Electrocardiography. Springfield, Illinois: Charles C. Thomas, Publisher, 1970.

16. Widrow, Bernard. "Adaptive Filters". In Aspects of Networks and System Theory, pp. 563-587. Edited by R.E. Kalman and N. DeClaris. New York: Holt, Rinehart and Winston, Inc., 1971.

17. Widrow, Bernard; Glover, John R.; McCool, John M.; Kaunitz, John; Williams, Charles S.; Hearn, Robert H.; Zeidler, James R.; Dong, Eugene, Jr.; and Goodlin, Robert C. "Adaptative Noise Cancelling: Principles and Applications." Proceedings of the IEEE 63, No. 12 (December 1975):1692-1716.

18. Widrow, Bernard; McCool, John M.; and Feintuch, Paul L. "Comments on 'An Adaptive Recursive LMS Filter'." Proceedings of the IEEE 65, No. 9 (September 1977):1402-1404.

19. Widrow, Bernard; McCool, John N.; Larimore, Michael G.; and Johnson, C. Richard, Jr. "Stationary and Nonstationary Learning Characteristics of the LMS Adaptive Filter." Proceedings of the IEEE 64, No. 8 (August 1976):1151-1162.

ACKNOWLEDGMENTS

I wish to acknowledge the efforts of Dr. David L. Carlson for his aid in defining the basic specification of the adaptive digital filter system and Dr. S. C. Dutta Roy for his clarification of many of the digital signal processing aspects of this system. A special thank you goes to Dr. Terry A. Smay who, aside from serving as my major professor, provided constructive criticism when this project progressed well and plenty of encouragement when progress seemed to be negative.

APPENDIX 1: SIMULATION PROGRAM LISTINGS

```
SCAN:PROCEDURE OPTIONS(MAIN);
/*                                                  */
/*    ROBERT W. WALSTROM                            */
/*    MAY 30, 1979                                  */
/*    THIS PROGRAM SCANS THE FREQUENCY              */
/*    SPECTRUM OF INTEREST AND                      */
/*    DETERMINES THE WEIGHT FACTORS FOR             */
/*    A FOURTH ORDER DIGITAL BANDPASS               */
/*    FILTER WITH A GIVEN FILTER                    */
/*    BANDWIDTH.                                    */
/*                                                  */
  DECLARE
    PI FLOAT(16) INITIAL(3.141592654),
    (FACTOR1,FACTOR2,FACTOR3,FACTOR4,FACTOR5,FACTOR6,FACTOR7,
      FREQ,BW,RATE,T,ONE,TWO,THREE,FOUR,SIX,EIGHT,
      TWELVE,K,PSI) FLOAT(16),
    (F1(90),F2(90),ALPHA(90)) FLOAT(16) INITIAL((90)0),
    (HEX1,HEX2,HEX3,HEX4,HEX5,HEX6,HEX7) CHAR(4),
    (I,J,L)  FIXED INITIAL(0);
  DECLARE HEX ENTRY(FLOAT(16)) RETURNS(CHAR(4));
  ONE=1.;
  TWO=2.;
  THREE=3.;
  FOUR=4.;
  SIX=6.;
  EIGHT=8.;
  TWELVE=12.;
  GET LIST(BW,RATE);
  PUT PAGE LIST('FOURTH ORDER DIGITAL BANDPASS FILTER');
  PUT SKIP(3) EDIT('BANDWIDTH = ',BW,' HZ')(A(12),F(3),A(3));
  PUT SKIP EDIT('SAMPLING RATE = ',RATE,' HZ')(A(16),F(3),A(3));
  T=ONE/RATE;
  FREQ=0;
  DO WHILE(BW/2>FREQ);
    I=I+2;
```

```
   FREQ=I*2;
END;
PSI=BW/(TWO*FREQ);
F1(I)=FREQ*(SQRT(PSI**2+ONE)-PSI);
F2(I)=FREQ*(SQRT(PSI**2+ONE)+PSI);
K=ONE/TAN(PI*(F2(I)-F1(I))*T);
PUT SKIP EDIT('K =',K)(A(3),F(8,4));
J=I;
DO WHILE(F2(J)<(RATE/TWO));
   ALPHA(J)=COS(PI*(F2(J)+F1(J))*T)/COS(PI*(F2(J)-F1(J))*T);
   J=J+1;
   FREQ=J*2;
   PSI=BW/(TWO*FREQ);
   F1(J)=FREQ*(SQRT(PSI**2+ONE)-PSI);
   F2(J)=FREQ*(SQRT(PSI**2+ONE)+PSI);
END;
J=J-1;
FACTOR5=ONE/(K**2+SQRT(TWO)*K+ONE);
HEX5=HEX(FACTOR5);
FACTOR6=-TWO*FACTOR5;
HEX6=HEX(FACTOR6);
FACTOR7=FACTOR5;
HEX7=HEX5;
FACTOR4=-(K**2-SQRT(TWO)*K+ONE)*FACTOR5;
HEX4=HEX(FACTOR4);
PUT SKIP(2) EDIT('FACTOR5 =',FACTOR5,'HEX5 = ',HEX5,'FACTOR6 =',
            FACTOR6,'HEX6 = ',HEX6,'FACTOR7 =',FACTOR7,'HEX7 = ',HEX7,
            'FACTOR4 =',FACTOR4,'HEX4 = ',HEX4)(4(A(9),F(7,4),X(1),
            A(7),A(4),X(3)));
PUT SKIP(3) EDIT('CENTER','UPPER','LOWER','ALPHA','FACTOR1','HEX1',
            'FACTOR2','HEX2','FACTOR3','HEX3')(A(6),3(X(3),A(5)),3(
            X(3),A(7),X(3),A(4)));
L=I;
DO WHILE(L<=J);
   FREQ=L*2;
```

```
      FACTOR3=TWO*ALPHA(L)*K*(TWO*K-SQRT(TWO))*FACTOR5;
      HEX3=HEX(FACTOR3);
      FACTOR2=-TWO*(TWO*(ALPHA(L)*K)**2+K**2-ONE)*FACTOR5;
      HEX2=HEX(FACTOR2);
      FACTOR1=TWO*ALPHA(L)*K*(TWO*K+SQRT(TWO))*FACTOR5;
      HEX1=HEX(FACTOR1);
      PUT SKIP EDIT(FREQ,F2(L),F1(L),ALPHA(L),FACTOR1,HEX1,FACTOR2,HEX2,
         FACTOR3,HEX3)(X(1),F(3),X(6),F(3),X(5),F(3),X(3),F(7,4),X(2),
         3(F(7,4),X(3),A(4),X(3)));
      L=L+1;
   END;
HEX:PROCEDURE(XFACTOR) RETURNS(CHAR(4));
/* THIS PROCEDURE CONVERTS FACTORS FROM DECIMAL TO HEXADECIMAL*/
/* REPRESENTATION.                                            */
   DECLARE
      HEXS CHARACTER(16) INITIAL('0123456789ABCDEF'),
      HEXNUM CHARACTER(4) INITIAL(''),
      (POS0,POS1,POS2,POS3) FIXED(2) INITIAL(0),
      (FACTOR,XFACTOR) FLOAT(16),
      NEG BIT(1) INITIAL('0'B);
   FACTOR=XFACTOR;
   IF FACTOR<0 THEN DO;
      NEG='1'B;
      FACTOR=ABS(FACTOR);
      END;
   POS1=FACTOR;
   FACTOR=FACTOR-POS1;
   FACTOR=FACTOR*16.;
   POS2=FACTOR;
   FACTOR=FACTOR-POS2;
   FACTOR=FACTOR*16.;
   POS3=FACTOR;
   FACTOR=FACTOR-POS3;
   POS4=FACTOR*16.;
   IF POS4>=8 THEN DO;
```

```
   POS3=POS3+1;
   IF POS3>=16 THEN DO;
     POS3=0;
     POS2=POS2+1;
     IF POS2>=16 THEN DO;
       POS2=0;
       POS1=POS1+1;
       IF POS1>=16 THEN DO;
         POS1=0;
         POS0=POS0+1;
         IF POS0>=16 THEN POS0=0;
       END;
     END;
   END;
END;
IF NEG THEN DO;
   POS3=15-POS3+1;
   POS2=15-POS2;
   POS1=15-POS1;
   POS0=15-POS0;
   IF POS3>=16 THEN DO;
     POS3=0;
     POS2=POS2+1;
     IF POS2>=16 THEN DO;
       POS2=0;
       POS1=POS1+1;
       IF POS1>=16 THEN DO;
         POS1=0;
         POS0=POS0+1;
         IF POS0>=16 THEN POS0=0;
       END;
     END;
   END;
END;
HEXNUM=SUBSTR(HEXS,POS0+1,1)||SUBSTR(HEXS,POS1+1,1)||SUBSTR(HEXS,
```

```
                    POS2+1,1)||SUBSTR(HEXS,POS3+1,1);
  RETURN (HEXNUM);
END HEX;
END SCAN;
```

```
FILTER:PROCEDURE OPTIONS(MAIN);
/*                                                    */
/*    ROBERT W. WALSTROM                              */
/*    NOVEMBER 22, 1978                               */
/*    2ND ORDER DIGITAL BANDPASS FILTER               */
/*    TIME RESPONSE ANALYSIS.                         */
/*                                                    */
   DECLARE
     PI FLOAT(16) INITIAL(3.141592654),
     (W2,W1,K,ALPHA,T,FACTOR1,FACTOR2,FACTOR3,FACTOR4,FACTOR5,FACTOR6,
       FACTOR7,FREQUENCY,FILTERIN,FILTEROUT,ONE,TWO,SUM) FLOAT(16),
     DELAY(4) FLOAT(16) INITIAL((4)0),
     (UPPER,LOWER,RATE,I,J,F)  FIXED;
   ONE=1.;
   TWO=2.;
   GET LIST(UPPER,LOWER,RATE);
   PUT SKIP LIST('UPPER CUTOFF FREQUENCY =',UPPER,'HERTZ');
   PUT SKIP LIST('LOWER CUTOFF FREQUENCY =',LOWER,'HERTZ');
   W2=TWO*UPPER*PI;
   PUT SKIP LIST('W2=',W2);
   W1=TWO*LOWER*PI;
   PUT SKIP LIST('W1=',W1);
   T=ONE/(TWO*RATE);
   PUT SKIP LIST('T=',T);
   ALPHA=COS((W2+W1)*T/TWO)/COS((W2-W1)*T/TWO);
   PUT SKIP LIST('ALPHA=',ALPHA);
   K=ONE/TAN((W2-W1)*T/TWO);
   PUT SKIP LIST('K=',K);
   FACTOR1=ONE/(K*K+SQRT(TWO)*K+ONE);
   PUT SKIP LIST('FACTOR1 =',FACTOR1);
   FACTOR2=-TWO*FACTOR1;
   PUT SKIP LIST('FACTOR2 =',FACTOR2);
   FACTOR3=FACTOR1;
   PUT SKIP LIST('FACTOR3 =',FACTOR3);
   FACTOR4=-(K*K-SQRT(TWO)*K+ONE)*FACTOR1;
```

```
   PUT SKIP LIST('FACTOR4 =',FACTOR4);
   FACTOR5=TWO*ALPHA*K*(TWO*K-SQRT(TWO))*FACTOR1;
   PUT SKIP LIST('FACTOR5 =',FACTOR5);
   FACTOR6=-TWO*(TWO*ALPHA**2*K*K+K*K-ONE)*FACTOR1;
   PUT SKIP LIST('FACTOR6 =',FACTOR6);
   FACTOR7=TWO*ALPHA*K*(TWO*K+SQRT(TWO))*FACTOR1;
   PUT SKIP LIST('FACTOR7 =',FACTOR7);
   DO F=0 BY 10 TO 180;
     FREQUENCY=FREQUENCY+10.;
     PUT PAGE EDIT('    FREQUENCY = ',FREQUENCY,' HZ') (A(17),F(3),A(3));
     PUT SKIP LIST('INTERVAL INPUT OUTPUT DELAY1 DELAY2 DELAY3 DELAY4');
     DO I=1 TO 100;
       FILTERIN=SIN(TWO*PI*FREQUENCY*I*T);
       SUM=FILTERIN+FACTOR7*DELAY(1)+FACTOR6*DELAY(2)+DELAY(3)*FACTOR5
         +DELAY(4)*FACTOR4;
       FILTEROUT=SUM*FACTOR1+DELAY(2)*FACTOR2+DELAY(4)*FACTOR3;
       PUT SKIP LIST(I,FILTERIN,FILTEROUT,DELAY);
       DO J=0 TO 2;
         DELAY(4-J)=DELAY(3-J);
       END;
       DELAY(1)=SUM;
     END;
     DELAY=0;
   END;
END FILTER;
```

```
RESPONS:PROCEDURE OPTIONS(MAIN);
/*                                                    */
/*    ROBERT W. WALSTROM                              */
/*    NOVEMBER 13, 1978                               */
/*    2ND ORDER DIGITAL BANDPASS FILTER               */
/*    FREQUENCY RESPONSE ANALYSIS.                    */
/*                                                    */
  DECLARE
    PI FLOAT(16) INITIAL(3.141592654),
    (W2,W1,K,ALPHA,GAIN,SUB1,SUB2,TRANSFER,T,N1,N2,N3,N4,D1,D2,D3,D4,
          MAG,FACTOR,ANGLE1,ANGLE2,ANGLE3,ANGLE4,ONE,TWO) FLOAT(16),
    (POLE1,POLE2,POLE3,POLE4,FREQ) FLOAT(16) COMPLEX,
    (UPPER,LOWER,F,RATE)  FIXED;
  GET LIST(UPPER,LOWER,RATE);
  PUT SKIP LIST('UPPER CUTOFF FREQUENCY =',UPPER,'HERTZ');
  PUT SKIP LIST('LOWER CUTOFF FREQUENCY =',LOWER,'HERTZ');
  ONE=1.;
  TWO=2.;
  W2=TWO*UPPER*PI;
  PUT SKIP LIST('W2=',W2,'RADIANS/SECOND');
  W1=TWO*LOWER*PI;
  PUT SKIP LIST('W1=',W1,'RADIANS/SECOND');
  T=ONE/(TWO*RATE);
  PUT SKIP LIST('T=',T,'SECONDS');
  ALPHA=COS((W2+W1)*T/TWO)/COS((W2-W1)*T/TWO);
  PUT SKIP LIST('ALPHA=',ALPHA);
  K=ONE/TAN((W2-W1)*T/TWO);
  PUT SKIP LIST('K=',K);
  GAIN=(K**2-SQRT(TWO)*K+ONE);
  PUT SKIP LIST('GAIN=',GAIN);
  FACTOR=K**2*(ALPHA**2-ONE);
  PUT SKIP LIST('FACTOR=',FACTOR);
  MAG=SQRT(SQRT(K**4*(ALPHA**2-ONE)**2+ONE));
  PUT SKIP LIST('MAG=',MAG);
  ANGLE1=ATAN(-ONE,FACTOR)/TWO;
```

```
PUT SKIP LIST('ANGLE1 =',ANGLE1);
ANGLE2=ANGLE1+PI;
PUT SKIP LIST('ANGLE2 =',ANGLE2);
ANGLE3=ATAN(ONE,FACTOR)/TWO;
PUT SKIP LIST('ANGLE3 =',ANGLE3);
ANGLE4=ANGLE3+PI;
PUT SKIP LIST('ANGLE4 =',ANGLE4);
POLE1=COMPLEX(SQRT(TWO)*((ALPHA*K+MAG*COS(ANGLE1))*(SQRT(TWO)*K-ONE)+
   MAG*SIN(ANGLE1))/(GAIN*TWO),SQRT(TWO)*(-(ALPHA*K+MAG*COS(ANGLE1))+
   (SQRT(TWO)*K-ONE)*MAG*SIN(ANGLE1))/(GAIN*TWO));
PUT SKIP LIST('POLE1 =',REAL(POLE1),'J',IMAG(POLE1));
POLE2=COMPLEX(SQRT(TWO)*((ALPHA*K+MAG*COS(ANGLE2))*(SQRT(TWO)*K-ONE)
   +MAG*SIN(ANGLE2))/(GAIN*TWO),SQRT(TWO)*(-(ALPHA*K+MAG*COS(ANGLE2))+
   (SQRT(TWO)*K-ONE)*MAG*SIN(ANGLE2))/(GAIN*TWO));
PUT SKIP LIST('POLE2 =',REAL(POLE2),'J',IMAG(POLE2));
POLE3=COMPLEX(SQRT(TWO)*((ALPHA*K+MAG*COS(ANGLE3))*(SQRT(TWO)*K-ONE)
   -MAG*SIN(ANGLE3))/(GAIN*TWO),SQRT(TWO)*(ALPHA*K+MAG*COS(ANGLE3)+
   (SQRT(TWO)*K-1.)*MAG*SIN(ANGLE3))/(GAIN*TWO));
PUT SKIP LIST('POLE3 =',REAL(POLE3),'J',IMAG(POLE3));
POLE4=COMPLEX(SQRT(TWO)*((ALPHA*K+MAG*COS(ANGLE4))*(SQRT(TWO)*K-ONE)
   -MAG*SIN(ANGLE4))/(GAIN*TWO),SQRT(TWO)*(ALPHA*K+MAG*COS(ANGLE4)+
   (SQRT(TWO)*K-ONE)*MAG*SIN(ANGLE4))/(GAIN*TWO));
PUT SKIP LIST('POLE4 =',REAL(POLE4),'J',IMAG(POLE4));
PUT PAGE;
DO F=5 TO (RATE-5) BY 5;
   FREQ=COMPLEX(COS(TWO*PI*F*T),SIN(TWO*PI*F*T));
   N1=SQRT((ONE-REAL(FREQ))**2+(IMAG(FREQ))**2);
   N2=SQRT((-ONE-REAL(FREQ))**2+(IMAG(FREQ))**2);
   N3=N2;
   N4=N1;
   D1=SQRT(((REAL(POLE1)-REAL(FREQ))**2)+(IMAG(POLE1)-IMAG(FREQ))**2);
   D2=SQRT(((REAL(POLE2)-REAL(FREQ))**2)+(IMAG(POLE2)-IMAG(FREQ))**2);
   D3=SQRT(((REAL(POLE3)-REAL(FREQ))**2)+(IMAG(POLE3)-IMAG(FREQ))**2);
   D4=SQRT(((REAL(POLE4)-REAL(FREQ))**2)+(IMAG(POLE4)-IMAG(FREQ))**2);
   TRANSFER=N1*N2*N3*N4/(D1*D2*D3*D4*GAIN);
```

```
      TRANSFER=20.*LOG10(TRANSFER);
      PUT SKIP EDIT(F,TRANSFER) (F(3),X(5),F(12,4));
    END;
END RESPONS;
```

APPENDIX 2: HARDWARE MULTIPLIER DESCRIPTION

General Description

The digital multiplier board provides high speed multiplication and product accumulation functions for any 8 bit microprocessor system configured around the Intel MULTIBUS (7). The board function requires fourteen memory locations and can be memory mapped into any 16 location memory block which is bounded by low order address digits of 0_H and F_H.

Circuit Description

The digital multiplier is configured around the TRW TDC1010J multiplier-accumulator integrated circuit. The TDC1010J contains two 16 bit input registers and a 35 bit accumulator/output buffer. The accumulator can be pre-loaded through the output buffer with any 35 bit two's complement or unsigned magnitude operand. The TDC1010J is capable of multiplying two 16 bit two's complement or unsigned magnitude operands and, if desired, adding the product to or subtracting the product from the accumulator, typically in 115 nanoseconds. It is possible to round off the lower order 16 bits of the accumulator value, if desired. The circuit diagram of the digital multiplier board is shown in Figure 20.

The digital multiplier board is selected by the 12 highest order address lines of the MULTIBUS. Each of these address lines is fed into one input of an exclusive NOR gate. The other exclusive NOR input is pulled high by a pull-up resistor while a switch is available to pull the input low. When the switch is open a high address signal is will cause

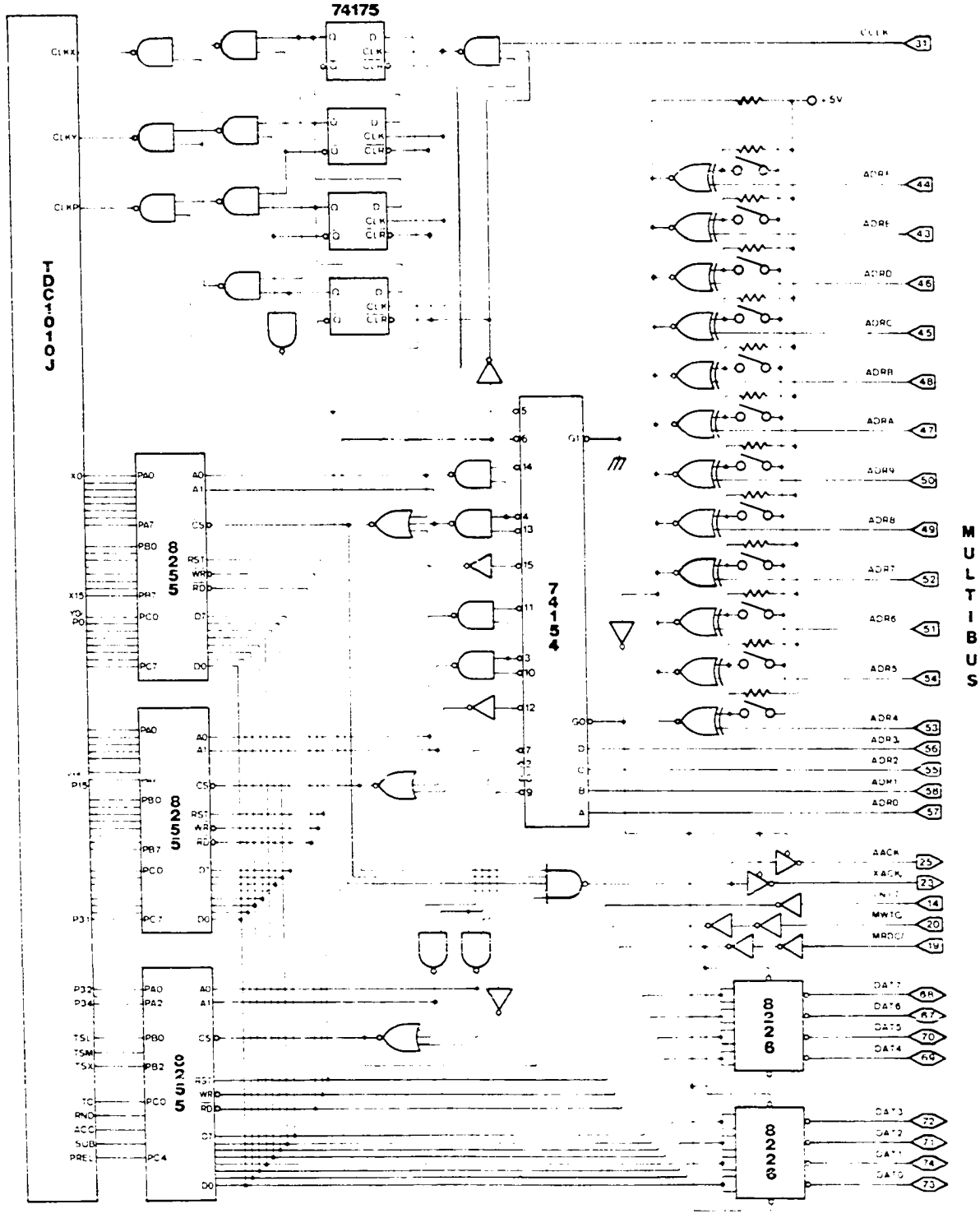


Figure 20. Hardware multiplier schematic diagram

the exclusive NOR gate output to go low. In like manner, when the switch is closed, an active low address signal will cause a high output on the exclusive NOR gate and a high address signal will cause a low exclusive NOR gate output.

The twelve exclusive NOR gates are connected in a wired-OR configuration. The digital multiplier board is then selected when the twelve high order address line bits correspond to the appropriate switch positions causing all twelve exclusive NOR gates to be in the high output state. Since these gates are in a wired-OR configuration, the wired-OR output is high. This high wired-OR output is inverted and selects the 74154 low order address decoder, enables the two 8226 bi-directional inverting data bus drivers, and activates the XACK/ and AACK/ bus drivers.

The 12 high order address lines select the digital multiplier board function and the 4 lower order address lines select the proper data and control memory locations of the board. These memory locations are shown in Figure 21.

Intel 8255 programmable peripheral interface chips (8) are used as data buffers between the MULTIBUS and the TDC1010J. Three 8255s are operated in Mode 1 (the basic input/output mode). Since, in this configuration, the digital multiplier board is interfaced with an 8 bit CPU, the data and control bytes are loaded in the 8255s 8 bits at a time. When the appropriate data and control bytes are loaded into the 8255s, the data and control bytes are transferred from the 8255 parallel output ports to the TDC1010J when the appropriate clock signals are applied to the TDC1010J.

Mnemonic	Location	Function
XLO	$XXX0_H$	Low order multiplier byte
XHI	$XXX1_H$	High order multiplier byte
YPLO1	$XXX2_H$	Low order multiplicand byte/low order byte of low order product word
YPLO2	$XXX3_H$	High order multiplicand byte / high order byte of low order product word
PHI1	$XXX4_H$	Low order byte of high order product word
PHI2	$XXX5_H$	High order byte of high order product word
PX	$XXX6_H$	Extended product byte
TSRCON	$XXX7_H$	TDC1010J buffer mode control register
MULCON	$XXX8_H$	TDC1010J mode control register
LDMULT	$XXX9_H$	Multiplication initiation control
LDPROD	$XXXA_H$	Product/accumulator preload control
CONT1	$XXXB_H$	A Buffer control register
CONT2	$XXXC_H$	B Buffer control register
CONT3	$XXXD_H$	C Buffer control register

Note: X's in the location address indicates a hardware switch settable value.

Figure 21. Multiplier memory locations

When the LDMULT memory location on the digital multiplier board is written to or read from, a control sequence is started that transfers the X and Y input data from the 8255 parallel ports into TDC1010J, multiplies these input values, and loads the product value from the TDC1010J accumulator into the TDC1010J output buffers. This control sequence is generated by a moebius or Johnson counter whose clock input is the CCLK/ constant clock signal from the MULTIBUS. A timing diagram of this sequence is shown in Figure 22. Note that the various TDC1010J functions are activated by the rising edges of the appropriate clock signals (CLKX, CLKY, CLKP). The counter is designed to halt after the CLKP signal has gone low and to reset when the digital multiplier board is deselected. The control sequence allows sufficient time for the input operands to be properly loaded and the multiplication to be performed before the product value is shifted to the TDC1010J output buffers. In addition, the control sequence will be complete within one machine cycle so that the system CPU will not enter any WAIT states before proceeding.

The rising CLKP signal necessary to pre-load the TDC1010J accumulator is obtained by inverting the 74154 active low signal which is generated when the LDPROD memory location is written to or read from.

Data are passed to and from the MULTIBUS through two Intel 8226 bi-directional inverting data bus drivers. The outputs from the 8226s are in the high impedance state until the digital multiplier board is selected. When the board is selected, the data are normally input from the MULTIBUS. When MRDC/ is active low, data are output from the 8226s to the MULTIBUS.

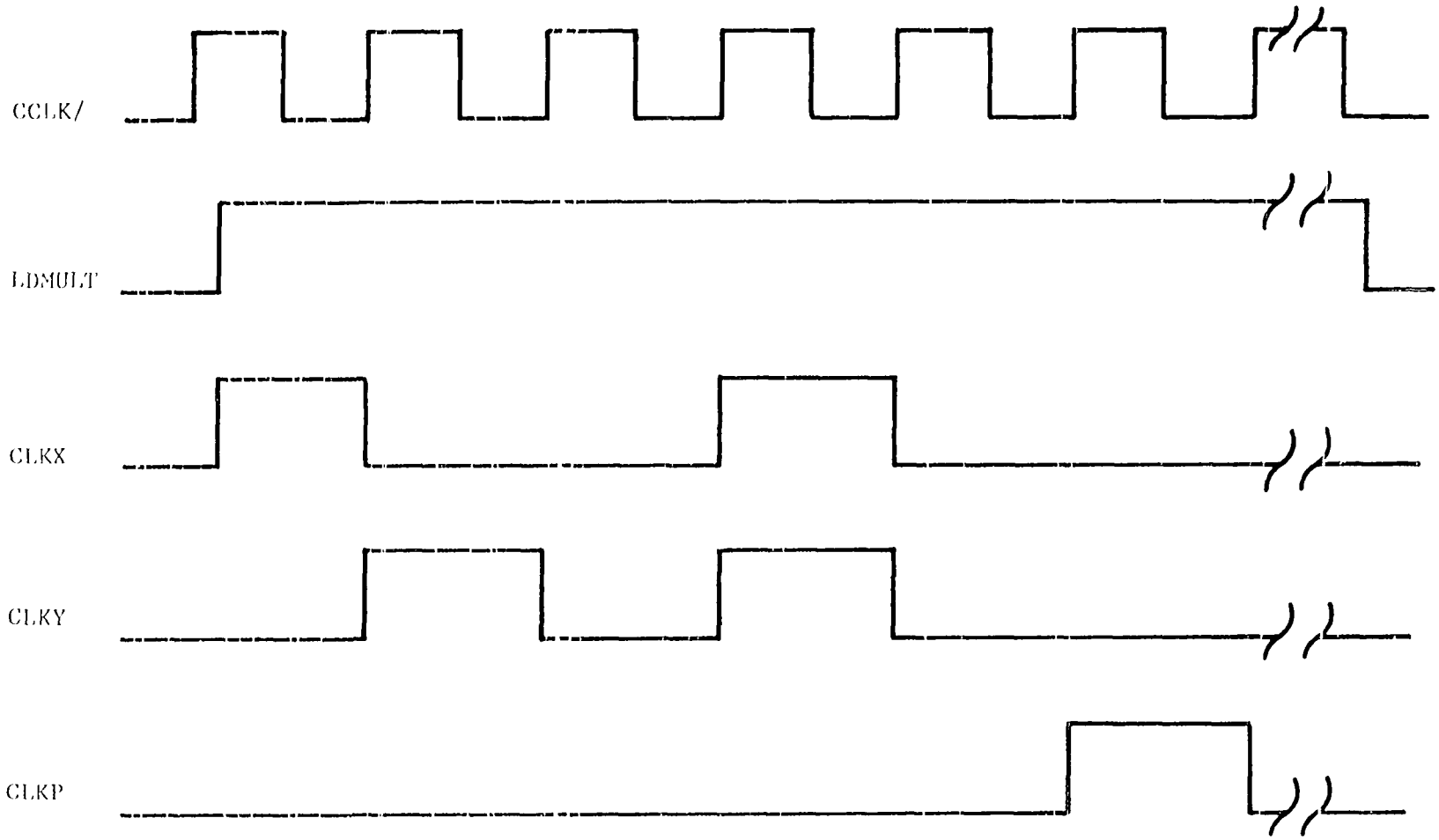


Figure 22. Multiplier sequence timing diagram

When the digital multiplier board is selected, the XACK/ and AACK/ buffers are connected to their appropriate lines on the MULTIBUS. These signals go active low when any of the digital multiplier board addresses are accessed. These two signals inform the system CPU that the address lines will be stable when the MRDC/ or MWTC/ signals are applied.

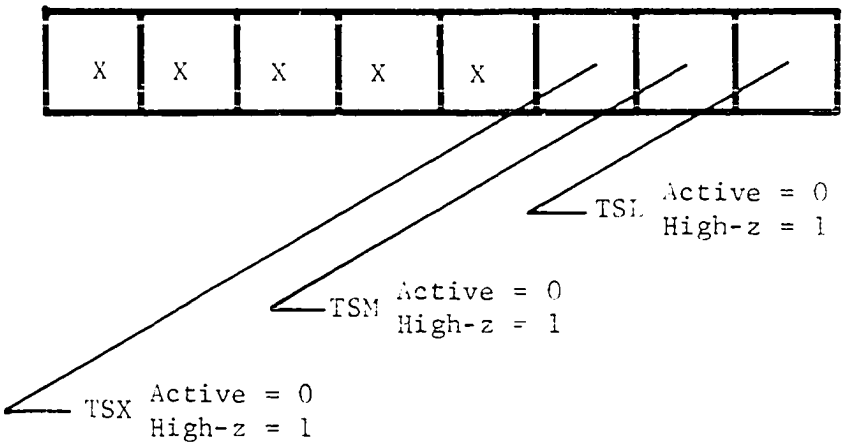
Operation

When power is applied to the digital multiplier board, the parallel ports and TDC1010J data and control ports are configured as inputs. The initial step in the operation of the digital multiplier board is to set the 8255s to Mode 1 and to configure the parallel ports as outputs. This is done by writing 80_H to the three 8255 control registers (CONT1, CONT2, CONT3).

Next, the TDC1010J output buffers are configured as appropriate by writing to the control register TSRCON. Figure 23 shows the format of the TSRCON control register.

The TDC1010J operation options are selected by writing an appropriate control word to the multiplier control register MULCON. Figure 24 shows the format of the MULCON multiplier control register.

To multiply two 16 bit two's complement operands and read the product, 00_H is placed in TSRCON and 01_H in MULCON. These control words set the TDC1010J output buffers to the high impedance state and direct the TDC1010J to treat the operands as two's complement values. The two's complement operands are read into the X and Y memory locations and any



TSL - Low Order Accumulator Word Register

TSM - High Order Accumulator Word Register

TSX - Extended Accumulator 3 Bit Register

Figure 23. TSRCON control register

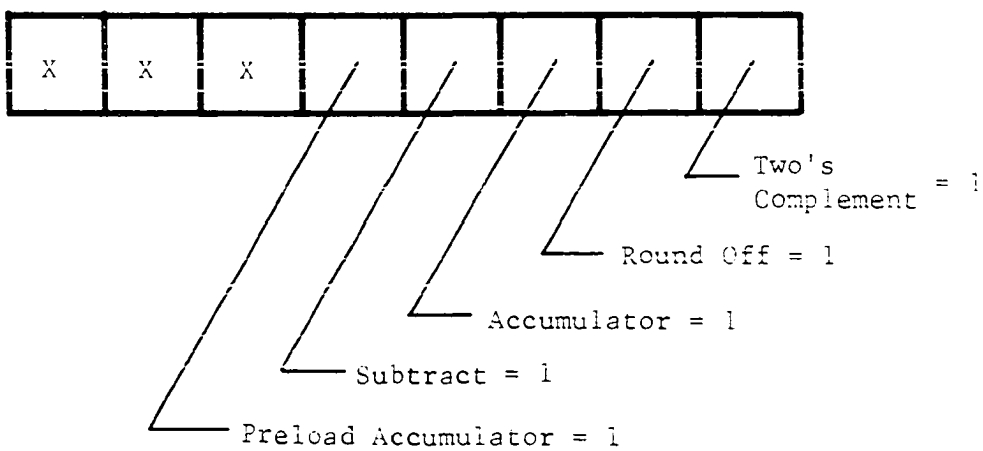


Figure 24. MULCON control register

value written to or read from the LDMULT memory location transfers the input values to the TDC1010J, initiates the multiplication, and moves the product value from the TDC1010J accumulator to the output buffers. Next, four of the 8255 parallel ports are reconfigured as inputs by placing 89_H and $9B_H$ into the 8255 control registers CONT1 and CONT2. (Note that when the 8255 control registers are altered, the parallel port buffers are cleared.) The TDC1010J output control registers are then placed on the appropriate pins by placing 03_H into TSRCON. Finally, the product value can be read from memory locations YPLO, YPHI, PLO, PHI and PX.

Other operation options may be selected by writing the appropriate values into MULCON. These options are two's complement or unsigned magnitude operands, adding or subtracting the product to or from the value in the TDC1010J accumulator, and rounding the product off to the seventeenth bit.

To pre-load a value into the TDC1010J accumulator the following steps are observed. Initialize the 8255 parallel ports as outlined before. Then configure the TDC1010J output buffers as inputs by writing 03_H into TSRCON and 10_H into MULCON. Write the data to be pre-loaded into YPLO, YPHI, PLO, PHI, and, if desired, PX. Finally, read or write any value to LDPROD. The TDC1010J accumulator is now preloaded.

APPENDIX 3: FILTER SYSTEM PROGRAM LISTING

RTI-1200 LOCATIONS

```
 LOC  OBJ          SEQ          SOURCE STATEMENT

                     1 ;*****************************************************
                     2 ;*****************************************************
                     3 ;**                                                 **
                     4 ;**           BAND PASS DIGITAL FILTER              **
                     5 ;**            INITIALIZATION ROUTINE               **
                     6 ;**       A/D-D/A BOARD FUNCTION ADDRESSES          **
                     7 ;**                                                 **
                     8 ;*****************************************************
                     9 ;*****************************************************
                    10 ;
                    11 ;                  VERSION 1, JANUARY 22, 1979
                    12 ;
                    13           NAME    ADDA
                    14           PUBLIC  CRDSEL,ADCHI,ADCLO,STATUS,CNVCMD
                    15           PUBLIC  MUXADR,GNSEL,ADC8,SETUP,DRIVE
                    16           PUBLIC  DAC1HI,DAC1LO,DAC2HI,DAC2LO
                    17 ;
                    18 ;
                    19 ;
EFFF                20 CRDSEL    EQU     0EFFFH
EFFE                21 ADCHI     EQU     0EFFEH
EFFD                22 ADCLO     EQU     0EFFDH
EFFC                23 STATUS    EQU     0EFFCH
```

RTI-1200 LOCATIONS

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  EFFB                24 CNVCMD  EQU     0EFFBH
  EFFA                25 MUXADR  EQU     0EFFAH
  EFF9                26 GNSEL   EQU     0EFF9H
  EFF8                27 ADC8    EQU     0EFF8H
  EFF7                28 DAC1HI  EQU     0EFF7H
  EFF6                29 DAC1LO  EQU     0EFF6H
  EFF5                30 DAC2HI  EQU     0EFF5H
  EFF4                31 DAC2LO  EQU     0EFF4H
  EFF3                32 DRIVE   EQU     0EFF3H
  EFF0                33 SETUP   EQU     0EFF0H
                      34 ;
                      35 ;
                      36         END
```

ASSEMBLY COMPLETE, NO ERRORS

MULTIPLIER LOCATIONS

```
LOC  OBJ          SEQ          SOURCE STATEMENT

                     1 ;*****************************************************
                     2 ;*****************************************************
                     3 ;**                                                 **
                     4 ;**          BAND PASS DIGITAL FILTER               **
                     5 ;**            INITIALIZATION ROUTINE               **
                     6 ;**         MULTIPLIER BOARD ADDRESSES              **
                     7 ;**                                                 **
                     8 ;*****************************************************
                     9 ;*****************************************************
                    10 ;
                    11 ;                VERSION 1, MARCH 27, 1979
                    12 ;
                    13          NAME    MULT
                    14          PUBLIC  X,YPLO,PHI,PX,PROD,TSRCON,MULCON
                    15          PUBLIC  CONT1,CONT2,CONT3,ADAPT,LDMULT,LDPROD
                    16 ;
                    17 ;
                    18 ;
                    19 ;
                    20 ;
F400                21 X        EQU     0F400H
F402                22 YPLO     EQU     0F402H
F403                23 PROD     EQU     0F403H
```

MULTIPLIER LOCATIONS

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  F404                24 PHI      EQU     0F404H
  F406                25 PX       EQU     0F406H
  F407                26 TSRCON   EQU     0F407H
  F408                27 MULCON   EQU     0F408H
  F409                28 LDMULT   EQU     0F409H
  F40A                29 LDPROD   EQU     0F40AH
  F40B                30 CONT1    EQU     0F40BH
  F40C                31 CONT2    EQU     0F40CH
  F40D                32 CONT3    EQU     0F40DH
  F40F                33 ADAPT    EQU     0F40FH
                      34 ;
                      35 ;
                      36 ;
                      37 ;
                      38
                      39          END

ASSEMBLY COMPLETE,  NO ERRORS
```

FILTER PARAMETERS

```
 LOC  OBJ          SEQ          SOURCE STATEMENT

                     1 ;*************************************************
                     2 ;*************************************************
                     3 ;**                                             **
                     4 ;**           BAND PASS DIGITAL FILTER          **
                     5 ;**          FOURTH ORDER WEIGHT VALUES         **
                     6 ;**                                             **
                     7 ;*************************************************
                     8 ;*************************************************
                     9 ;
                    10 ;                 VERSION 3, JUNE 2, 1979
                    11 ;
                    12           NAME    ORD4TH
                    13           PUBLIC  FACT1,FACT2,FACT3,FACT4,FACT5,FACT6,FACT7
                    14           PUBLIC  WGHT1,WGHT2,WGHT3,WGHT4,WGHT5,WGHT6,WGHT7
                    15 ;
                    16 ;
                    17           DSEG
0000 FBFF           18 WGHT1:    DW      0FFFBH
0002 5EFE           19 WGHT2:    DW      0FE5EH
0004 FCFF           20 WGHT3:    DW      0FFFCH
0006 4FFF           21 WGHT4:    DW      0FF4FH
0008 0400           22 WGHT5:    DW      0004H
000A F9FF           23 WGHT6:    DW      0FFF9H
```

FILTER PARAMETERS

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  000C 0400           24 WGHT7:  DW       0004H
  0002                25 FACT1:  DS       2
  0002                26 FACT2:  DS       2
  0002                27 FACT3:  DS       2
  0002                28 FACT4:  DS       2
  0002                29 FACT5:  DS       2
  0002                30 FACT6:  DS       2
  0002                31 FACT7:  DS       2
                      32 ;
                      33         END
```

ASSEMBLY COMPLETE, NO ERRORS

```
LOC  OBJ          SEQ          SOURCE STATEMENT

                     1 ;*****************************************************
                     2 ;*****************************************************
                     3 ;**                                                 **
                     4 ;**           BAND PASS DIGITAL FILTER              **
                     5 ;**               SYSTEM VARIABLES                  **
                     6 ;**                                                 **
                     7 ;*****************************************************
                     8 ;*****************************************************
                     9 ;
                    10 ;                 VERSION 4, JUNE 7, 1979
                    11 ;
                    12           NAME    VALUES
                    13           PUBLIC  DELAY1,DELAY2,DELAY3,DELAY4,DELAY5
                    14           PUBLIC  NXTSTP,THRESH,PEAK,LEFT,RIGHT
                    15           PUBLIC  COUNT,NODE,BRANCH,VAFT1,VAFT2,VAFT3
                    16           PUBLIC  NOWNOD,SCALE,SCALER
                    17 ;
                    18 ;
                    19 ;
                    20 ;
                    21 ;
                    22           DSEG
                    23 ;
```

FILTER VARIABLES

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  0002                24 DELAY1:  DS      2
  0002                25 DELAY2:  DS      2
  0002                26 DELAY3:  DS      2
  0002                27 DELAY4:  DS      2
  0002                28 DELAY5:  DS      2
  0002                29 LEFT:    DS      2
  0002                30 RIGHT:   DS      2
  0002                31 PEAK:    DS      2
  0002                32 THRESH:  DS      2
  0002                33 SCALER:  DS      2
  0002                34 NXTSTP:  DS      2
  0001                35 NOWNOD:  DS      1
  0001                36 SCALE:   DS      1
  0001                37 COUNT:   DS      1
  0001                38 NODE:    DS      1
  0001                39 BRANCH:  DS      1
  0001                40 VAFT1:   DS      1
  0001                41 VAFT2:   DS      1
  0001                42 VAFT3:   DS      1
                      43 ;
                      44 ;
                      45 ;
                      46 ;
                      47          END

ASSEMBLY COMPLETE,  NO ERRORS
```

LOOK-UP TABLES

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

                      1 ;**********************************************************
                      2 ;**********************************************************
                      3 ;**                                                      **
                      4 ;**           BAND PASS DIGITAL FILTER                   **
                      5 ;**        ADAPTIVE ROUTINE LOOK-UP TABLES               **
                      6 ;**                                                      **
                      7 ;**********************************************************
                      8 ;**********************************************************
                      9 ;
                     10 ;                  VERSION 4, JUNE 8, 1979
                     11 ;
                     12            NAME    LOOKUP
                     13 ;
                     14 ;
                     15 ;**       BRANCH TABLE - THIS TABLE CONTAINS THE   **
                     16 ;**       LOW ORDER ADDRESS LOCATIONS FOR THE      **
                     17 ;**       BINARY TREE STRUCTURES.                  **
                     18 ;
                     19 ;
                     20 ;
                     21            PUBLIC  B40,F140,F240,F340,S40,T40
                     22 ;
                     23 ;
```

LOOK-UP TABLES

```
LOC  OBJ          SEQ          SOURCE STATEMENT

6030              24           ORG     6030H
                  25 ;
                  26 ;
6030 FFFF         27 B40:      DW      0FFFFH
6032 FFFF         28 B50:      DW      0FFFFH,0FFFFH,0H,363CH,0FFFFH
6034 FFFF
6036 0000
6038 3C36
603A FFFF
603C 3E3A         29 B60:      DW      3A3EH,0FFFFH,0H,384CH,0FFFFH
603E FFFF
6040 0000
6042 4C38
6044 FFFF
6046 4844         30 B70:      DW      4448H,0FFFFH,0H,4650H,0FFFFH
6048 FFFF
604A 0000
604C 5046
604E FFFF
6050 524E         31 B80:      DW      4E52H,0FFFFH,0H,426AH,0FFFFH
6052 FFFF
6054 0000
6056 6A42
6058 FFFF
605A 5C58         32 B90:      DW      585CH,0FFFFH,0H,5A64H,0FFFFH
605C FFFF
```

LOOK-UP TABLES

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

  605E 0000
  6060 645A
  6062 FFFF
  6064 6662            33 B1FF:   DW      6266H,0FFFFH,0H,6074H,0FFFFH
  6066 FFFF
  6068 0000
  606A 7460
  606C FFFF
  606E 706C            34 B110:   DW      6C70H,0FFFFH,0H,6E7AH,0FFFFH
  6070 FFFF
  6072 0000
  6074 7A6E
  6076 FFFF
  6078 0000            35 B120:   DW      0H,767CH,0FFFFH,0H,56A8H
  607A 7C76
  607C FFFF
  607E 0000
  6080 A856
  6082 FFFF            36 B130:   DW      0FFFFH,8286H,0FFFFH,0H,848EH
  6084 8682
  6086 FFFF
  6088 0000
  608A 8E84
  608C FFFF            37 B140:   DW      0FFFFH,8C90H,0FFFFH,0H,8A9EH
  608E 908C
  6090 FFFF
```

LOOK-UP TABLES

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  6092 0000
  6094 9E8A
  6096 FFFF           38 B150:   DW      0FFFFH,969AH,0FFFFH,0H,98A2H
  6098 9A96
  609A FFFF
  609C 0000
  609E A298
  60A0 FFFF           39 B160:   DW      0FFFFH,0A0A4H,0FFFFH,0H,94BCH
  60A2 A4A0
  60A4 FFFF
  60A6 0000
  60A8 BC94
  60AA FFFF           40 B170:   DW      0FFFFH,0AAAEH,0FFFFH,0H,0ACB6H
  60AC AEAA
  60AE FFFF
  60B0 0000
  60B2 B6AC
  60B4 FFFF           41 B180:   DW      0FFFFH,0B4B8H,0FFFFH,0H,0B2C6H
  60B6 B8B4
  60B8 FFFF
  60BA 0000
  60BC C6B2
  60BE FFFF           42 B190:   DW      0FFFFH,0BEC2H,0FFFFH,0H,0C0C8H
  60C0 C2BE
  60C2 FFFF
  60C4 0000
```

LOOK-UP TABLES

```
LOC  OBJ          SEQ          SOURCE STATEMENT

60C6 C8C0
60C8 FFFF           43 B200:   DW      0FFFFH,0FFFFH,0H,0FFFFH
60CA FFFF
60CC 0000
60CE FFFF
                    44 ;
                    45 ;
                    46 ;
                    47 ;
                    48 ;**     VFAT1 TABLE - THIS TABLE CONTAINS THE   **
                    49 ;**     THE APPROPRIATE FILTER FACTOR1 WEIGHT   **
                    50 ;**     VALUES TO BE USED IN THE ADAPTIVE       **
                    51 ;**     ROUTINE.                                **
                    52 ;
                    53 ;
612E                54         ORG     612EH
                    55 ;
612E 9903           56 F140:   DW      0399H,0394H
6130 9403
6132 8E03           57 F150:   DW      038EH,0386H,037EH,0374H,036AH
6134 8603
6136 7E03
6138 7403
613A 6A03
613C 5E03           58 F160:   DW      035EH,0351H,0343H,0335H,0325H
613E 5103
```

LOOK-UP TABLES

```
  LOC  OBJ           SEQ           SOURCE STATEMENT

  6140 4303
  6142 3503
  6144 2503
  6146 1403            59 F170:    DW        0314H,0302H,02EFH,02DCH,02C7H
  6148 0203
  614A EF02
  614C DC02
  614E C702
  6150 B202            60 F180:    DW        02B2H,029CH,0284H,026DH,0254H
  6152 9C02
  6154 8402
  6156 6D02
  6158 5402
  615A 3B02            61 F190:    DW        023BH,0220H,0206H,01EAH,01CEH
  615C 2002
  615E 0602
  6160 EA01
  6162 CE01
  6164 B201            62 F1100:   DW        01B2H,0195H,0177H,0159H,013BH
  6166 9501
  6168 7701
  616A 5901
  616C 3B01
  616E 1C01            63 F1110:   DW        011CH,00FDH,00DDH,00BDH,009DH
  6170 FD00
  6172 DD00
```

LOOK-UP TABLES

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  6174 BD00
  6176 9D00
  6178 7D00           64 F1120:  DW       007DH,005DH,003CH,001CH,0FFFBH
  617A 5D00
  617C 3C00
  617E 1C00
  6180 FBFF
  6182 DAFF           65 F1130:  DW       0FFDAH,0FFBAH,0FF99H,0FF79H,0FF59H
  6184 BAFF
  6186 99FF
  6188 79FF
  618A 59FF
  618C 39FF           66 F1140:  DW       0FF39H,0FF19H,0FEF9H,0FEDAH,0FEBBH
  618E 19FF
  6190 F9FE
  6192 DAFE
  6194 BBFE
  6196 9DFE           67 F1150:  DW       0FE9DH,0FE7FH,0FE61H,0FE44H,0FE28H
  6198 7FFE
  619A 61FE
  619C 44FE
  619E 28FE
  61A0 0CFE           68 F1160:  DW       0FE0CH,0FDF0H,0FDD6H,0FDBCH,0FDA2H
  61A2 F0FD
  61A4 D6FD
  61A6 BCFD
```

LOOK-UP TABLES

```
LOC  OBJ           SEQ          SOURCE STATEMENT

61A8 A2FD
61AA 8AFD            69 F1170:  DW       0FD8AH,0FD72H,0FD5BH,0FD45H,0FD2FH
61AC 72FD
61AE 5BFD
61B0 45FD
61B2 2FFD
61B4 1BFD            70 F1180:  DW       0FD1BH,0FD07H,0FCF5H,0FCE3H,0FCD2H
61B6 07FD
61B8 F5FC
61BA E3FC
61BC D2FC
61BE C2FC            71 F1190:  DW       0FCC2H,0FCB4H,0FCA6H,0FC99H,0FC8EH
61C0 B4FC
61C2 A6FC
61C4 99FC
61C6 8EFC
61C8 83FC            72 F1200:  DW       0FC83H,0FC79H,0FC71H,0FC6AH,0FC64H
61CA 79FC
61CC 71FC
61CE 6AFC
61D0 64FC
61D2 5FFC            73 F1210:  DW       0FC5FH
                     74 ;
                     75 ;
                     76 ;**     VAFT2 TABLE - THIS TABLE CONTAINS ALL   **
                     77 ;**     OF THE FACT2 WEIGHT VALUES USED BY THE  **
```

LOOK-UP TABLES

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

                      78 ;**      ADAPTIVE ROUTINE.                                **
                      79 ;
                      80 ;
  622E                81          ORG     622EH
                      82 ;
  622E 1CFB           83 F240:    DW      0FB1CH,0FB25H
  6230 25FB
  6232 30FB           84 F250:    DW      0FB30H,0FB3DH,0FB4CH,0FB5DH,0FB6FH
  6234 3DFB
  6236 4CFB
  6238 5DFB
  623A 6FFB
  623C 83FB           85 F260:    DW      0FB83H,0FB99H,0FBB0H,0FBC8H,0FBE1H
  623E 99FB
  6240 B0FB
  6242 C8FB
  6244 E1FB
  6246 FBFB           86 F270:    DW      0FBFBH,0FC17H,0FC33H,0FC4FH,0FC6CH
  6248 17FC
  624A 33FC
  624C 4FFC
  624E 6CFC
  6250 8AFC           87 F280:    DW      0FC8AH,0FCA8H,0FCC5H,0FCE3H,0FD0AH
  6252 A8FC
  6254 C5FC
  6256 E3FC
```

LOOK-UP TABLES

```
LOC  OBJ           SEQ          SOURCE STATEMENT

6258 0AFD
625A 1EFD            88 F290:   DW      0FD1EH,0FD3AH,0FD56H,0FD71H,0FD8CH
625C 3AFD
625E 56FD
6260 71FD
6262 8CFD
6264 A5FD            89 F2100:  DW      0FDA5H,0FDBDH,0FDD3H,0FDE9H,0FDFDH
6266 BDFD
6268 D3FD
626A E9FD
626C FDFD
626E 0FFE            90 F2110:  DW      0FE0FH,0FE1FH,0FE2EH,0FE3BH,0FE46H
6270 1FFE
6272 2EFE
6274 3BFE
6276 46FE
6278 4FFE            91 F2120:  DW      0FE4FH,0FE56H,0FE5AH,0FE5DH,0FE5EH
627A 56FE
627C 5AFE
627E 5DFE
6280 5EFE
6282 5DFF            92 F2130:  DW      0FE5DH,0FE59H,0FE54H,0FE4CH,0FE42H
6284 59FE
6286 54FE
6288 4CFE
628A 42FE
```

LOOK-UP TABLES

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  628C 37FE           93 F2140:  DW       0FE37H,0FE29H,0FE1AH,0FE09H,0FDF6H
  628E 29FE
  6290 1AFE
  6292 09FE
  6294 F6FD
  6296 E2FD           94 F2150:  DW       0FDE2H,0FDCCH,0FDB5H,0FD9CH,0FD82H
  6298 CCFD
  629A B5FD
  629C 9CFD
  629E 82FD
  62A0 68FD           95 F2160:  DW       0FD68H,0FD4CH,0FD30H,0FD13H,0FCF5H
  62A2 4CFD
  62A4 30FD
  62A6 13FD
  62A8 F5FC
  62AA D7FC           96 F2170:  DW       0FCD7H,0FCB9H,0FC9BH,0FC7DH,0FC5FH
  62AC B9FC
  62AE 9BFC
  62B0 7DFC
  62B2 5FFC
  62B4 42FC           97 F2180:  DW       0FC42H,0FC25H,0FC08H,0FBEBH,0FBD2H
  62B6 25FC
  62B8 08FC
  62BA EBFB
  62BC D2FB
  62BE B9FB           98 F2190:  DW       0FBB9H,0FBA1H,0FB8AH,0FB74H,0FB60H
```

LOOK-UP TABLES

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  62C0 A1FB
  62C2 8AFB
  62C4 74FB
  62C6 60FB
  62C8 4DFB           99 F2200:  DW       0FB4DH,0FB3DH,0FB2EH,0FD21H,0FB16H
  62CA 3DFB
  62CC 2EFB
  62CE 21FB
  62D0 16FB
  62D2 0CFB          100 F2210:  DW       0FB0CH
                     101 ;
                     102 ;
                     103 ;
                     104 ;**     VAFT3 TABLE - THIS TABLE CONTAINS ALL    **
                     105 ;**     OF THE FACT3 FILTER WEIGHT VALUES FOR    **
                     106 ;**     USE BY THE ADAPTIVE ROUTINE.             **
                     107 ;
                     108 ;
  632E               109          ORG      632EH
                     110 ;
  632E FC02          111 F340:    DW       02FCH,02F8H
  6330 F802
  6332 F302          112 F350:    DW       02F3H,02EDH,0236H,02DEH,0D5H
  6334 ED02
  6336 3602
  6338 DE02
```

LOOK-UP TABLES

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  633A D500
  633C CB02          113 F360:    DW       02CBH,02C1H,02B5H,02A9H,029CH
  633E C102
  6340 B502
  6342 A902
  6344 9C02
  6346 8E02          114 F370:    DW       028EH,027FH,026FH,025FH,024EH
  6348 7F02
  634A 6F02
  634C 5F02
  634E 4E02
  6350 3C02          115 F380:    DW       023CH,022AH,0217H,0203H,01EEH
  6352 2A02
  6354 1702
  6356 0302
  6358 EE01
  635A D901          116 F390:    DW       01D9H,01C4H,01AEH,0197H,0180H
  635C C401
  635E AE01
  6360 9701
  6362 8001
  6364 6801          117 F3100:   DW       0168H,0150H,0137H,011EH,0105H
  6366 5001
  6368 3701
  636A 1E01
  636C 0501
```

LOOK-UP TABLES

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  636E EB00          118 F3110:  DW      00EBH,00D2H,00B7H,009DH,0082H
  6370 D200
  6372 B700
  6374 9D00
  6376 8200
  6378 6800          119 F3120:  DW      0068H,004DH,0032H,0017H,0FFFCH
  637A 4D00
  637C 3200
  637E 1700
  6380 FCFF
  6382 E1FF          120 F3130:  DW      0FFE1H,0FFC6H,0FFABH,0FF90H,0FF75H
  6384 C6FF
  6386 ABFF
  6388 90FF
  638A 75FF
  638C 5BFF          121 F3140:  DW      0FF5BH,0FF40H,0FF26H,0FF0CH,0FEF3H
  638E 40FF
  6390 26FF
  6392 0CFF
  6394 F3FE
  6396 D9FE          122 F3150:  DW      0FED9H,0FEC0H,0FEA8H,0FE90H,0FE78H
  6398 C0FE
  639A A8FE
  639C 90FE
  639E 78FE
  63A0 61FE          123 F3160:  DW      0FE61H,0FE4AH,0FE34H,0FE1EH,0FE09H
```

LOOK-UP TABLES

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  63A2 4AFE
  63A4 34FE
  63A6 1EFE
  63A8 09FE
  63AA F5FD          124 F3170:  DW      OFDF5H,OFDE1H,OFDCEH,OFDBCH,OFDAAH
  63AC E1FD
  63AE CEFD
  63B0 BCFD
  63B2 AAFD
  63B4 99FD          125 F3180:  DW      OFD99H,OFD89H,OFD79H,OFD6BH,OFD5DH
  63B6 89FD
  63B8 79FD
  63BA 6BFD
  63BC 5DFD
  63BE 50FD          126 F3190:  DW      OFD50H,OFD43H,OFD38H,OFD2DH,OFD24H
  63C0 43FD
  63C2 38FD
  63C4 2DFD
  63C6 24FD
  63C8 1BFD          127 F3200:  DW      OFD1BH,OFD13H,OFD0CH,OFD06H,OFD01H
  63CA 13FD
  63CC 0CFD
  63CE 06FD
  63D0 01FD
  63D2 FDFC          128 F3210:  DW      OFCFDH
                     129 ;
```

LOOK-UP TABLES

```
LOC  OBJ           SEQ          SOURCE STATEMENT

                    130 ;
                    131 ;
                    132 ;**     SCALER TABLE - THIS TABLE CONTAINS ALL  **
                    133 ;**     OF THE INPUT SCALER VALUES FOR USE BY   **
                    134 ;**     THE ADAPTIVE AND BASIC FILTER ROUTINES. **
                    135 ;
                    136 ;
6430                137          ORG     6430H
                    138 ;
6430 1000           139 S40:     DW      0010H
6432 0000           140 S50:     DW      0H,0001H,0010H,0010H,0010H
6434 0100
6436 1000
6438 1000
643A 1000
643C 1400           141 S60:     DW      0014H,0020H,0024H,0030H,0034H
643E 2000
6440 2400
6442 3000
6444 3400
6446 3C00           142 S70:     DW      003CH,0048H,0048H,004CH,0058H
6448 4800
644A 4800
644C 4C00
644E 5800
6450 6400           143 S80:     DW      0064H,007CH,0080H,008CH,0094H
```

LOOK-UP TABLES

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

  6452 7C00
  6454 8000
  6456 8C00
  6458 9400
  645A 9C00          144 S90:     DW       009CH,00A8H,00CCH,00BCH,00D8H
  645C A800
  645E CC00
  6460 BC00
  6462 D800
  6464 D800          145 S100:    DW       00D8H,00E4H,00ECH,00ECH,00F8H
  6466 E400
  6468 EC00
  646A EC00
  646C F800
  646E F400          146 S110:    DW       00F4H,0100H,0100H,0100H,0100H
  6470 0001
  6472 0001
  6474 0001
  6476 0001
  6478 0001          147 S120:    DW       0100H,0100H,0100H,0100H,0100H
  647A 0001
  647C 0001
  647E 0001
  6480 0001
  6482 0001          148 S130:    DW       0100H,0100H,0100H,0100H,0100H
  6484 0001
```

LOOK-UP TABLES

```
LOC  OBJ           SEQ          SOURCE STATEMENT

6486 0001
6488 0001
648A 0001
648C 0001          149 S140:    DW      0100H,0100H,0100H,0100H,00E8H
648E 0001
6490 0001
6492 0001
6494 E800
6496 F000          150 S150:    DW      00F0H,00E8H,00E0H,00C8H,00C0H
6498 E800
649A E000
649C C800
649E C000
64A0 BC00          151 S160:    DW      00BCH,00B4H,00ACH,00ACH,009CH
64A2 B400
64A4 AC00
64A6 AC00
64A8 9C00
64AA 8800          152 S170:    DW      0088H,008DH,0078H,0064H,0060H
64AC 8D00
64AE 7800
64B0 6400
64B2 6000
64B4 5800          153 S180:    DW      0058H,004CH,0040H,0034H,0028H
64B6 4C00
64B8 4000
```

LOOK-UP TABLES

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

  64BA 3400
  64BC 2800
  64BE 2400          154 S190:    DW      0024H,0028H,001CH,0018H,000BH
  64C0 2800
  64C2 1C00
  64C4 1800
  64C6 0B00
  64C8 0700          155 S200:    DW      0007H
                     156 ;
                     157 ;
                     158 ;
                     159 ;**      THRESHOLD TABLE - THIS TABLE CONTAINS   **
                     160 ;**      ALL OF THE APPLICABLE THRESHOLD VALUES  **
                     161 ;**      FOR USE IN THE ADAPTIVE ROUTINE.        **
                     162 ;
                     163 ;
  6530               164          ORG     6530H
                     165 ;
  6530 FFFF          166 T40:     DW      0FFFFH
  6532 FFFF          167 T50:     DW      0FFFFH,0FFFFH,0FFFFH,0FFDCH,0FFFFH
  6534 FFFF
  6536 FFFF
  6538 DCFF
  653A FFFF
  653C A8FF          168 T60:     DW      0FFA8H,0FFFFH,0FFFFH,0FF44H,0FFFFH
  653E FFFF
```

LOOK-UP TABLES

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  6540 FFFF
  6542 44FF
  6544 FFFF
  6546 D0FE          169 T70:    DW      0FED0H,0FFFFH,0FFFFH,0FE2AH,0FFFFH
  6548 FFFF
  654A FFFF
  654C 2AFE
  654E FFFF
  6550 A3FD          170 T80:    DW      0FDA3H,0FFFFH,0FFFFH,0FCB6H,0FFFFH
  6552 FFFF
  6554 FFFF
  6556 B6FC
  6558 FFFF
  655A 54FC          171 T90:    DW      0FC54H,0FFFFH,0FFFFH,0FB28H,0FFFFH
  655C FFFF
  655E FFFF
  6560 28FB
  6562 FFFF
  6564 D0FA          172 T100:   DW      0FAD0H,0FFFFH,0FFFFH,0FA20H,0FFFFH
  6566 FFFF
  6568 FFFF
  656A 20FA
  656C FFFF
  656E F8F9          173 T110:   DW      0F9F8H,0FFFFH,0FFFFH,0F9B0H,0FFFFH
  6570 FFFF
  6572 FFFF
```

LOOK-UP TABLES

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  6574 80F9
  6576 FFFF
  6578 FFFF          174 T120:   DW      0FFFFH,0F9CCH,0FFFFH,0FFFFH,0F99CH
  657A CCF9
  657C FFFF
  657E FFFF
  6580 9CF9
  6582 FFFF          175 T130:   DW      0FFFFH,0F980H,0FFFFH,0FFFFH,0F950H
  6584 80F9
  6586 FFFF
  6588 FFFF
  658A 50F9
  658C FFFF          176 T140:   DW      0FFFFH,0F960H,0FFFFH,0FFFFH,0FA60H
  658E 60F9
  6590 FFFF
  6592 FFFF
  6594 60FA
  6596 FFFF          177 T150:   DW      0FFFFH,0FA3CH,0FFFFH,0FFFFH,0FAF0H
  6598 3CFA
  659A FFFF
  659C FFFF
  659E F0FA
  65A0 FFFF          178 T160:   DW      0FFFFH,0FB9CH,0FFFFH,0FFFFH,0FC5CH
  65A2 9CFB
  65A4 FFFF
  65A6 FFFF
```

LOOK-UP TABLES

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

  65A8 5CFC
  65AA FFFF          179 T170:    DW      0FFFFH,0FC90H,0FFFFH,0FFFFH,0FDDCH
  65AC 90FC
  65AE FFFF
  65B0 FFFF
  65B2 DCFD
  65B4 FFFF          180 T180:    DW      0FFFFH,0FE6CH,0FFFFH,0FFFFH,0FF18H
  65B6 6CFE
  65B8 FFFF
  65BA FFFF
  65BC 18FF
  65BE FFFF          181 T190:    DW      0FFFFH,0FF4CH,0FFFFH,0FFFFH,0FFCCH
  65C0 4CFF
  65C2 FFFF
  65C4 FFFF
  65C6 CCFF
  65C8 FFFF          182 T200:    DW      0FFFFH
                     183 ;
                     184 ;
                     185 ;
                     186 ;
                     187 ;
                     188                  END

ASSEMBLY COMPLETE,  NO ERRORS
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

                     1 ;*****************************************************
                     2 ;*****************************************************
                     3 ;**                                                 **
                     4 ;**           BAND PASS DIGITAL FILTER              **
                     5 ;**            INITIALIZATION ROUTINE               **
                     6 ;**                                                 **
                     7 ;*****************************************************
                     8 ;*****************************************************
                     9 ;
                    10 ;                VERSION 8, JUNE 7, 1979
                    11 ;
                    12           NAME    INIT
                    13           EXTRN   CRDSEL,ADCHI,ADCLO,STATUS,CNVCMD
                    14           EXTRN   MUXADR,GNSEL,ADC8,DAC1HI,DAC1LO
                    15           EXTRN   DAC2HI,DAC2LO,DRIVE,SETUP,X,YPLO,PHI
                    16           EXTRN   PX,PROD,TSRCON,MULCON,LDMULT,LDPROD
                    17           EXTRN   CONT1,CONT2,CONT3,PEAK,COUNT,NODE
                    18           EXTRN   DELAY1,DELAY2,DELAY3,DELAY4,DELAY5
                    19           EXTRN   FILTER,STEP1,NXTSTP,RESET,B40,F140
                    20           EXTRN   F240,F340,BRANCH,VAFT1,VAFT2,VAFT3
                    21           EXTRN   THRESH,NOWNOD,SCALE,SCALER,S40,T40
                    22           PUBLIC  INIT
                    23 ;
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

                   24 ;
                   25 ;
                   26 ;**      INITIALIZE THE APPROPRIATE INTERRUPT    **
                   27 ;**      VECTORS.                                **
                   28 ;
                   29          ASEG
                   30 ;
                   31 ;**      RESTART 3 IS TRIGGERED BY THE RTI-1200  **
                   32 ;**      R-C PACER CLOCK TO START THE FILTER     **
                   33 ;**      ROUTINE.                                **
                   34 ;
0018               35          ORG     18H
0018 C30000    E   36          JMP     FILTER
                   37 ;
                   38 ;
                   39 ;**      RESTART 4 IS TRIGGERED FROM THE FRONT   **
                   40 ;**      PANEL AND REINITIALIZES THE FILTER      **
                   41 ;**      GAINS TO THEIR INITIAL VALUES.          **
                   42 ;
0020               43          ORG     20H
0020 C30000    E   44          JMP     RESET
                   45 ;
                   46 ;
                   47 ;**      INITIALIZE THE FILTER.                  **
                   48 ;
                   49          CSEG
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

                   50 ;
0000 F3            51 INIT:    DI
                   52 ;
                   53 ;
                   54 ;**      PRESET THE INTERRUPT MASK.                **
                   55 ;
0001 3E12          56          MVI     A,12H
0003 D3FD          57          OUT     0FDH
0005 3E00          58          MVI     A,0H
0007 D3FC          59          OUT     0FCH
0009 3EE0          60          MVI     A,0E0H
000B D3FC          61          OUT     0FCH
                   62 ;
                   63 ;
                   64 ;**      INITIALIZE THE STACK POINTER.             **
                   65 ;
000D 310000    S   66          LXI     SP,STACK
                   67 ;
                   68 ;
                   69 ;**      INITIALIZE THE FILTER WEIGHTS.            **
                   70 ;
0010 E7            71          RST     4
                   72 ;
                   73 ;
                   74 ;**      INITIALIZE THE RTI-1200 BOARD FUNCTIONS.**
                   75 ;
```

INITIALIZATION ROUTINE

```
LOC  OBJ           SEQ          SOURCE STATEMENT

                    76 ;**      INITIALIZE THE D/A CONVERTER TO ZERO.    **
                    77 ;
0011 210000         78          LXI     H,0H
0014 220000    E    79          SHLD    DAC1LO
                    80 ;
                    81 ;
                    82 ;**      SET THE ANALOG INPUT GAIN TO 1.          **
                    83 ;
0017 3E00           84          MVI     A,0H
0019 320000    E    85          STA     GNSEL
                    86 ;
                    87 ;
                    88 ;**      SELECT THE THIRD ANALOG INPUT CHANNEL.   **
                    89 ;
001C 3E03           90          MVI     A,3H
001E 320000    E    91          STA     MUXADR
                    92 ;
                    93 ;
                    94 ;**      SET THE RTI-1200 'PACER' CLOCK SYSTEM    **
                    95 ;**      SO THAT THE INTERNAL R-C 'PACER' WILL    **
                    96 ;**      TRIGGER THE SYSTEM INTERRUPT 4.          **
                    97 ;**      THE R-C 'PACER' DETERMINES THE FILTER    **
                    98 ;**      SAMPLING INTERVAL.                       **
                    99 ;
0021 3E02          100          MVI     A,2H
0023 320000    E   101          STA     SETUP
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

                   102 ;
                   103 ;
                   104 ;**     INITIALIZE THE AVERAGE PEAK TO ZERO AND **
                   105 ;**     THE COUNT VALUE TO 24 FOR THE ADAPTIVE  **
                   106 ;**     FUNCTION.                               **
                   107 ;
0026 210000        108          LXI     H,0H
0029 220000    E   109          SHLD    PEAK
002C 3E18          110          MVI     A,24D
002E 320000    E   111          STA     COUNT
                   112 ;
                   113 ;
                   114 ;**     INITIALIZE THE OTHER ADAPTIVE FUNCTION  **
                   115 ;**     VALUES.                                 **
                   116 ;
0031 3E80          117          MVI     A,128D
0033 320000    E   118          STA     NOWNOD
0036 320000    E   119          STA     NODE
0039 210000    E   120          LXI     H,STEP1
003C 220000    E   121          SHLD    NXTSTP
                   122 ;
                   123 ;
                   124 ;**     LOCATE THE LOOK-UP TABLE LOCATIONS FOR  **
                   125 ;**     USE IN THE ADAPTIVE ROUTINE.            **
                   126 ;
003F 210000    E   127          LXI     H,B40
```

INITIALIZATION ROUTINE

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

  0042 7C            128          MOV     A,H
  0043 320000    E   129          STA     BRANCH
  0046 210000    E   130          LXI     H,F140
  0049 7C            131          MOV     A,H
  004A 320000    E   132          STA     VAFT1
  004D 210000    E   133          LXI     H,F240
  0050 7C            134          MOV     A,H
  0051 320000    E   135          STA     VAFT2
  0054 210000    E   136          LXI     H,F340
  0057 7C            137          MOV     A,H
  0058 320000    E   138          STA     VAFT3
  0058 210000    E   139          LXI     H,T40
  005E 7C            140          MOV     A,H
  005F 320000    E   141          STA     THRESH
  0062 210000    E   142          LXI     H,S40
  0065 7C            143          MOV     A,H
  0066 320000    E   144          STA     SCALE
                     145 ;
                     146 ;
                     147 ;**      PRESET THE INPUT SCALER VALUE.          **
                     148 ;
  0069 3A0000    E   149          LDA     NOWNOD
  006C 6F            150          MOV     L,A
  006D 5E            151          MOV     E,M
  006E 23            152          INX     H
  006F 56            153          MOV     D,M
```

INITIALIZATION ROUTINE

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

0070 EB              154           XCHG
0071 220000    E     155           SHLD    SCALER
                     156 ;
                     157 ;
                     158 ;**       INITIALIZE THE FILTER DELAY VALUES TO   **
                     159 ;**       ZERO.                                   **
                     160 ;
0074 160A            161           MVI     D,0AH
0076 010000    E     162           LXI     B,DELAY1
0079 3E00            163           MVI     A,0
007B 02              164 INDEL:    STAX    B
007C 03              165           INX     B
007D 15              166           DCR     D
007E C27B00    C     167           JNZ     INDEL
                     168 ;
                     169 ;
                     170 ;**       INITIALIZE THE TDC1010J MULTIPLIER      **
                     171 ;**       FUNCTION TO PRELOAD THE INPUT OPERAND   **
                     172 ;**       VALUES.                                 **
                     173 ;
                     174 ;**       INITIALIZE THE 8255 I/O REGISTERS AS    **
                     175 ;**       OUTPUTS.                                **
                     176 ;
0081 3E80            177           MVI     A,80H
0083 320000    E     178           STA     CONT1
0086 320000    E     179           STA     CONT2
```

INITIALIZATION ROUTINE

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  0089 320000   E    180          STA     CONT3
                     181 ;
                     182 ;
                     183 ;**      SET THE TDC1010J OUTPUT REGISTERS TO    **
                     184 ;**      HI-Z.                                   **
                     185 ;
  008C 3E07          186          MVI     A,07H
  008E 320000   E    187          STA     TSRCON
                     188 ;
                     189 ;
                     190 ;**      SET THE TDC1010J TO PRELOAD THE PRODUCT **
                     191 ;**      REGISTERS.                              **
                     192 ;
  0091 3E01          193          MVI     A,01H
  0093 320000   E    194          STA     MULCON
                     195 ;
                     196 ;
                     197 ;**      AWAIT THE START OF THE FILTER OPERATION.**
                     198 ;
  0096 FB            199 STDBY:   EI
  0097 C39600   C    200          JMP     STDBY
                     201 ;
                     202 ;
                     203          END

ASSEMBLY COMPLETE,  NO ERRORS
```

PARAMETER INITIALIZATION

```
LOC  OBJ          SEQ          SOURCE STATEMENT

                     1 ;*****************************************************
                     2 ;*****************************************************
                     3 ;**                                                 **
                     4 ;**           BAND PASS DIGITAL FILTER              **
                     5 ;**                RESTART ROUTINE                  **
                     6 ;**                                                 **
                     7 ;*****************************************************
                     8 ;*****************************************************
                     9 ;
                    10 ;                VERSION 1, MARCH 27, 1979
                    11 ;
                    12           NAME    RESET
                    13           PUBLIC  RESET
                    14           EXTRN   FACT1,WGHT1
                    15 ;
                    16 ;
                    17 ;**       THIS ROUTINE REINITIALIZES THE SECOND    **
                    18 ;**       ORDER FILTER WEIGHTS.                    **
                    19 ;
                    20 ;
                    21 ;
                    22           CSEG
0000 F3             23 RESET:    DI
```

PARAMETER INITIALIZATION

```
  LOC  OBJ           SEQ           SOURCE STATEMENT

  0001 210000    E    24            LXI     H,FACT1
  0004 EB             25            XCHG
  0005 210000    E    26            LXI     H,WGHT1
  0008 060E           27            MVI     B,0EH
  000A 7E             28 RELOAD:    MOV     A,M
  000B 12             29            STAX    D
  000C 23             30            INX     H
  000D 13             31            INX     D
  000E 05             32            DCR     B
  000F C20A00    C    33            JNZ     RELOAD
  0012 3E20           34            MVI     A,20H
  0014 D3FD           35            OUT     0FDH
  0016 FB             36            EI
  0017 C9             37            RET
                      38 ;
                      39 ;
                      40 ;
                      41            END
```

ASSEMBLY COMPLETE, NO ERRORS

BASIC FILTER

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

                      1 ;*****************************************************
                      2 ;*****************************************************
                      3 ;**                                                 **
                      4 ;**           BAND PASS DIGITAL FILTER              **
                      5 ;**                FOURTH ORDER                     **
                      6 ;**                                                 **
                      7 ;*****************************************************
                      8 ;*****************************************************
                      9 ;
                     10 ;        VERSION 5, JUNE 7, 1979
                     11 ;
                     12 ;**      THIS ROUTINE IS A FOURTH ORDER BUTTERWORTH  **
                     13 ;**      BANDPASS DIGITAL FILTER. THIS ROUTINE IS    **
                     14 ;**      INTENDED TO BE LINKED WITH AN ADAPTIVE      **
                     15 ;**      ROUTINE.                                    **
                     16 ;
                     17 ;
                     18          NAME    FILTER
                     19          EXTRN   FACT1,FACT2,FACT3,FACT4,FACT5,FACT6
                     20          EXTRN   FACT7,DELAY1,DELAY2,DELAY3,DELAY4
                     21          EXTRN   DELAY5,X,YPLO,PHI,PX,PROD,PEAK,COUNT
                     22          EXTRN   TSRCON,MULCON,LDMULT,LDPROD,CONT1
                     23          EXTRN   CONT2,CONT3,CNVCMD,ADCLO,STATUS
```

```
 LOC  OBJ          SEQ          SOURCE STATEMENT

                    24           EXTRN   ADCHI,DAC1LO,ADAPT,ADAPTR,SCALER
                    25           STKLN   10D
                    26 ;
                    27           CSEG
                    28           PUBLIC  FILTER
                    29 ;
                    30 ;**       THE MAIN FILTER ROUTINE.                    **
                    31 ;
                    32 ;
0000 F3             33 FILTER:   DI
                    34 ;
                    35 ;**       INITIATE THE A/D CONVERSION PROCESS.        **
                    36 ;
0001 320000    E    37           STA     CNVCMD
                    38 ;
                    39 ;
                    40 ;**       WHILE WAITING FOR THE A/D CONVERSION TO COMPLETE,**
                    41 ;**       LOAD THE POINTER TO THE MULTIPLY INITIATE    **
                    42 ;**       COMMAND LOCATION.                           **
                    43 ;
0004 010000    E    44           LXI     B,LDMULT
                    45 ;
                    46 ;
                    47 ;**       COMMENCE THE FORMATION OF THE COMPOSITE INPUT  **
                    48 ;**       VALUE TO BE MADE UP OF THE SUMS OF THE        **
                    49 ;**       PRODUCTS OF DELAYS 1-4 AND THE WEIGHT FACTORS  **
```

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

                      50 ;**      1-4.  THE DELAY VALUES ARE SHIFTED FOR          **
                      51 ;**      THE NEXT SAMPLED DATA PROCESS.  THIS            **
                      52 ;**      SHIFTING IS ACCOUNTED FOR WHEN THE OUTPUT       **
                      53 ;**      VALUE IS FORMED.                                **
                      54 ;
  0007 2A0000    E    55          LHLD    FACT1
  000A 220000    E    56          SHLD    X
  000D 2A0000    E    57          LHLD    DELAY1
  0010 220000    E    58          SHLD    YPLO
  0013 02             59          STAX    B
  0014 EB             60          XCHG
                      61 ;
                      62 ;
                      63 ;**      SET UP THE MULTIPLIER FUNCTION TO SUM THE       **
                      64 ;**      REMAINING PRODUCTS.                             **
                      65 ;
  0015 3E05           66          MVI     A,05H
  0017 320000    E    67          STA     MULCON
                      68 ;
  001A 2A0000    E    69          LHLD    FACT2
  001D 220000    E    70          SHLD    X
  0020 2A0000    E    71          LHLD    DELAY2
  0023 220000    E    72          SHLD    YPLO
  0026 02             73          STAX    B
  0027 EB             74          XCHG
  0028 220000    E    75          SHLD    DELAY2
```

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

                      76 ;
  002B 2A0000    E    77           LHLD    FACT3
  002E 220000    E    78           SHLD    X
  0031 2A0000    E    79           LHLD    DELAY3
  0034 220000    E    80           SHLD    YPLO
  0037 02             81           STAX    B
  0038 EB             82           XCHG
  0039 220000    E    83           SHLD    DELAY3
                      84 ;
  003C 2A0000    E    85           LHLD    FACT4
  003F 220000    E    86           SHLD    X
  0042 2A0000    E    87           LHLD    DELAY4
  0045 220000    E    88           SHLD    YPLO
  0048 02             89           STAX    B
  0049 220000    E    90           SHLD    DELAY5
  004C EB             91           XCHG
  004D 220000    E    92           SHLD    DELAY4
                      93 ;
  0050 2A0000    E    94           LHLD    SCALER
  0053 220000    E    95           SHLD    X
  0056 2A0000    E    96           LHLD    ADCLO
  0059 220000    E    97           SHLD    YPLO
  005C 02             98           STAX    B
                      99 ;
                     100 ;
                     101 ;**       SET THE MULTIPLIER CONTROL REGISTERS    **
```

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

                      102 ;**      TO OUTPUT THE COMPOSITE INPUT VALUE.    **
                      103 ;
  005D 3E9B           104          MVI     A,9BH
  005F 320000    E    105          STA     CONT2
  0062 3E00           106          MVI     A,0H
  0064 320000    E    107          STA     TSRCON
                      108 ;
                      109 ;
                      110 ;**      REMOVE THE COMPOSITE INPUT VALUE FROM   **
                      111 ;**      THE MULTIPLIER AND STORE THIS VALUE     **
                      112 ;**      IN THE H-L REGISTER PAIR                **
                      113 ;
  0067 2A0000    E    114          LHLD    PROD
                      115 ;
                      116 ;
                      117 ;**      RESET THE MULTIPLIER TO INPUT OPERANDS  **
                      118 ;**      AND FORM THE FILTER DIGITAL OUTPUT.     **
                      119 ;
  006A 220000    E    120          SHLD    DELAY1
  006D 3E01           121          MVI     A,01H
  006F 320000    E    122          STA     MULCON
  0072 3E07           123          MVI     A,07H
  0074 320000    E    124          STA     TSRCON
  0077 3E80           125          MVI     A,80H
  0079 320000    E    126          STA     CONT2
                      127 ;
```

```
 LOC  OBJ          SEQ          SOURCE STATEMENT

                    128 ;
                    129 ;**     FORM THE DIGITAL OUTPUT VALUE.          **
                    130 ;
007C 220000    E    131          SHLD    YPLO
007F 2A0000    E    132          LHLD    FACT5
0082 220000    E    133          SHLD    X
0085 02             134          STAX    B
0086 3E05           135          MVI     A,05H
0088 320000    E    136          STA     MULCON
                    137 ;
008B 2A0000    E    138          LHLD    DELAY3
008E 220000    E    139          SHLD    YPLO
0091 2A0000    E    140          LHLD    FACT6
0094 220000    E    141          SHLD    X
0097 02             142          STAX    B
                    143 ;
0098 2A0000    E    144          LHLD    DELAY5
009B 220000    E    145          SHLD    YPLO
009E 2A0000    E    146          LHLD    FACT7
00A1 220000    E    147          SHLD    X
00A4 02             148          STAX    B
                    149 ;
                    150 ;
                    151 ;**     SET UP THE MULTIPLIER CONTROL REGISTERS **
                    152 ;**     TO OUTPUT THE FILTER DIGITAL OUTPUT     **
                    153 ;
```

```
LOC  OBJ           SEQ          SOURCE STATEMENT

00A5 3E9B            154          MVI     A,9BH
00A7 320000    E     155          STA     CONT2
00AA 3E00            156          MVI     A,0H
00AC 320000    E     157          STA     TSRCON
                     158 ;
                     159 ;
                     160 ;**      FETCH THE FILTER DIGITAL OUTPUT VALUE   **
                     161 ;**      AND PASS THE VALUE TO THE D/A CONVERTER.**
                     162 ;
00AF 2A0000    E     163          LHLD    PROD
00B2 220000    E     164          SHLD    DAC1LO
                     165 ;
                     166 ;
                     167 ;**      DETERMINE IF THE NEW FILTER OUTPUT IS   **
                     168 ;**      THE LARGEST VALUE UP TO THIS POINT.  IF **
                     169 ;**      SO, REPLACE THE PEAK VALUE WITH THIS    **
                     170 ;**      LATEST FILTER OUTPUT VALUE.             **
                     171 ;
00B5 3A0000    E     172          LDA     COUNT
00B8 FE10            173          CPI     16D
00BA D2E000    C     174          JNC     SKIP
00BD EB              175          XCHG
00BE 2A0000    E     176          LHLD    PROD
00C1 29              177          DAD     H
00C2 D2CC00    C     178          JNC     PLUS
00C5 7B              179          MOV     A,E
```

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

  00C6 2F             180          CMA
  00C7 5F             181          MOV     E,A
  00C8 7A             182          MOV     A,D
  00C9 2F             183          CMA
  00CA 57             184          MOV     D,A
  00CB 13             185          INX     D
  00CC 2A0000    E    186 PLUS:    LHLD    PEAK
  00CF 7C             187          MOV     A,H
  00D0 BA             188          CMP     D
  00D1 DADC00    C    189          JC      SWITCH
  00D4 C2E000    C    190          JNZ     SKIP
  00D7 7D             191          MOV     A,L
  00D8 BB             192          CMP     E
  00D9 D2E000    C    193          JNC     SKIP
  00DC EB             194 SWITCH:  XCHG
  00DD 220000    E    195          SHLD    PEAK
                      196 ;
                      197 ;
                      198 ;**      RESET THE MULTIPLIER TO THE INPUT MODE. **
                      199 ;
  00E0 3E07           200 SKIP:    MVI     A,07H
  00E2 320000    E    201          STA     TSRCON
  00E5 3E01           202          MVI     A,01H
  00E7 320000    E    203          STA     MULCON
  00EA 3E80           204          MVI     A,80H
  00EC 320000    E    205          STA     CONT2
```

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

                     206 ;
                     207 ;
                     208 ;**     TEST TO SEE IF ADAPTOR FUNCTION IS TO   **
                     209 ;**     BE PERFORMED.                           **
                     210 ;
  00EF 3A0000    E   211          LDA      ADAPT
  00F2 17            212          RAL
  00F3 D20001    C   213          JNC      CONTIN
  00F6 3A0000    E   214          LDA      COUNT
  00F9 3D            215          DCR      A
  00FA CA0000    E   216          JZ       ADAPTR
  00FD 320000    E   217          STA      COUNT
  0100 3A0000    E   218 CONTIN:  LDA      STATUS
  0103 3E20          219          MVI      A,20H
  0105 D3FD          220          OUT      0FDH
  0107 FB            221          EI
  0108 C9            222          RET
                     223 ;
                     224 ;
                     225 ;
                     226 ;
                     227 ;
                     228 ;
                     229 ;
                     230          END

ASSEMBLY COMPLETE,  NO ERRORS
```

ADAPTIVE ROUTINE

```
 LOC  OBJ          SEQ          SOURCE STATEMENT

                     1 ;**********************************************************
                     2 ;**********************************************************
                     3 ;**                                                      **
                     4 ;**          ADAPTATION ROUTINE FOR THE                  **
                     5 ;**         FOURTH ORDER BANDPASS FILTER                 **
                     6 ;**                  ROUTINE                             **
                     7 ;**                                                      **
                     8 ;**********************************************************
                     9 ;**********************************************************
                    10 ;
                    11 ;         VERSION 6, JUNE 8, 1979
                    12 ;
                    13 ;
                    14 ;**       THIS ROUTINE IS TO BE LINKED TO A          **
                    15 ;**       FOURTH ORDER DIGITAL BANDPASS FILTER.      **
                    16 ;
                    17 ;
                    18           NAME    ADAPTOR
                    19           EXTRN   FACT1,FACT2,FACT3,NXTSTP,THRESH
                    20           EXTRN   STATUS,PEAK,COUNT,NODE,BRANCH
                    21           EXTRN   VAFT1,VAFT2,VAFT3,LEFT,RIGHT
                    22           EXTRN   DELAY1,DELAY2,DELAY3,DELAY4
                    23           EXTRN   NOWNOD,SCALE,SCALER
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

                   24          STKLN   10D
                   25          PUBLIC  ADAPTR,STEP1
                   26 ;
                   27 ;
                   28          CSEG
                   29 ;
                   30 ;
                   31 ;**      DETERMINE WHICH PART OF THIS ALGORITHM  **
                   32 ;**      IS TO BE USED.                          **
                   33 ;
0000 2A0000    E   34 ADAPTR: LHLD    NXTSTP
0003 E9            35          PCHL
                   36 ;
                   37 ;
                   38 ;**      LOCATE THE APPROPRIATE THRESHOLD VALUE. **
                   39 ;
0004 3A0000    E   40 STEP1:  LDA     NOWNOD
0007 6F            41          MOV     L,A
0008 3A0000    E   42          LDA     THRESH
000B 67            43          MOV     H,A
000C 5E            44          MOV     E,M
000D 23            45          INX     H
000E 56            46          MOV     D,M
                   47 ;
                   48 ;
                   49 ;**      COMPARE THE NODE OUTPUT PEAK VALUE WITH **
```

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

                     50 ;**      THE THRESHOLD VALUE.                          **
                     51 ;
000F 2A0000    E     52 STEP2:   LHLD    PEAK
0012 19              53          DAD     D
0013 29              54          DAD     H
                     55 ;
                     56 ;
                     57 ;**      RESET THE PEAK TO ZERO AND COUNT TO 24         **
                     58 ;**      TO BE READY FOR THE NEXT PEAK                 **
                     59 ;**      COMPUTATION.                                  **
                     60 ;
0014 210000          61          LXI     H,0
0017 220000    E     62          SHLD    PEAK
001A 3E18            63          MVI     A,24D
001C 320000    E     64          STA     COUNT
001F D29E01    C     65          JNC     RECHEK
                     66 ;
                     67 ;
                     68 ;**      LOAD THE RIGHT BRANCH FILTER WEIGHTS          **
                     69 ;**      INTO THE FILTER.                              **
                     70 ;
0022 3A0000    E     71 STEP3:   LDA     NODE
0025 6F              72          MOV     L,A
0026 3A0000    E     73          LDA     BRANCH
0029 67              74          MOV     H,A
002A 7E              75          MOV     A,M
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

002B 320000   E    76           STA     NOWNOD
                   77 ;
                   78 ;
                   79 ;**      LOAD THE RIGHT FACT1 VALUE INTO THE     **
                   80 ;**      FILTER.                                 **
                   81 ;
002E 6F            82           MOV     L,A
002F 3A0000   E    83           LDA     VAFT1
0032 67            84           MOV     H,A
0033 5E            85           MOV     E,M
0034 23            86           INX     H
0035 56            87           MOV     D,M
0036 2B            88           DCX     H
0037 EB            89           XCHG
0038 220000   E    90           SHLD    FACT1
003B EB            91           XCHG
                   92 ;
                   93 ;
                   94 ;**      LOAD THE RIGHT FACT2 VALUE INTO THE     **
                   95 ;**      FILTER.                                 **
                   96 ;
003C 3A0000   E    97           LDA     VAFT2
003F 67            98           MOV     H,A
0040 5E            99           MOV     E,M
0041 23           100           INX     H
0042 56           101           MOV     D,M
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

0043 2B           102          DCX     H
0044 EB           103          XCHG
0045 220000   E   104          SHLD    FACT2
0048 EB           105          XCHG
                  106 ;
                  107 ;
                  108 ;**      LOAD THE RIGHT FACT3 VALUE INTO THE     **
                  109 ;**      FILTER.                                 **
                  110 ;
0049 3A0000   E   111          LDA     VAFT3
004C 67           112          MOV     H,A
004D 5E           113          MOV     E,M
004E 23           114          INX     H
004F 56           115          MOV     D,M
0050 2B           116          DCX     H
0051 EB           117          XCHG
0052 220000   E   118          SHLD    FACT3
0055 EB           119          XCHG
                  120 ;
                  121 ;
                  122 ;**      LOAD THE INPUT VALUE SCALE FACTOR.      **
                  123 ;
0056 3A0000   E   124          LDA     SCALE
0059 67           125          MOV     H,A
005A 5E           126          MOV     E,M
005B 23           127          INX     H
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

005C 56           128          MOV     D,M
005D EB           129          XCHG
005E 220000   E   130          SHLD    SCALER
                  131 ;
                  132 ;
                  133 ;**      RESET THE DELAYED INPUTS TO ZERO.        **
                  134 ;
                  135 ;
0061 210000       136          LXI     H,0
0064 220000   E   137          SHLD    DELAY1
0067 220000   E   138          SHLD    DELAY2
006A 220000   E   139          SHLD    DELAY3
006D 220000   E   140          SHLD    DELAY4
                  141 ;
                  142 ;
                  143 ;**      SET THE POINTER TO THE NEXT PORTION      **
                  144 ;**      OF THE ALGORITHM TO BE OPERATED WHEN     **
                  145 ;**      CONTROL IS NEXT RETURNED TO THIS         **
                  146 ;**      ROUTINE.                                 **
                  147 ;
0070 217F00   C   148          LXI     H,STEP4
0073 220000   E   149          SHLD    NXTSTP
                  150 ;
                  151 ;
                  152 ;**      RESET THE MDS SYSTEM INTERRUPTS AND      **
                  153 ;**      THE RTI-1200 INTERRUPTS.                 **
```

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

                     154 ;
0076 3E20            155          MVI     A,20H
0078 D3FD            156          OUT     0FDH
007A 320000    E     157          STA     STATUS
007D FB              158          EI
007E C9              159          RET
                     160 ;
                     161 ;
                     162 ;**      LOAD THE APPROPRIATE THRESHOLD VALUE.    **
                     163 ;
007F 3A0000    E     164 STEP4:   LDA     NOWNOD
0082 6F              165          MOV     L,A
0083 3A0000    E     166          LDA     THRESH
0086 67              167          MOV     H,A
0087 5E              168          MOV     E,M
0088 23              169          INX     H
0089 56              170          MOV     D,M
                     171 ;
                     172 ;
                     173 ;**      COMPARE THE RIGHT OUTPUT PEAK VALUE      **
                     174 ;**      WITH THE THRESHOLD VALUE.                **
                     175 ;
008A 2A0000    E     176 STEP5:   LHLD    PEAK
008D 220000    E     177          SHLD    RIGHT
0090 19              178          DAD     D
0091 29              179          DAD     H
```

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

                      180 ;
                      181 ;
                      182 ;**     RESET THE PEAK TO ZERO AND THE COUNT TO          **
                      183 ;**     24 FOR THE NEXT CALCULATION OF THE      **
                      184 ;**     OUTPUT AVERAGE VALUE.                   **
                      185 ;
0092 210000           186          LXI     H,0
0095 220000     E     187          SHLD    PEAK
0098 3E18             188          MVI     A,24D
009A 320000     E     189          STA     COUNT
009D D29E01     C     190          JNC     RECHEK
                      191 ;
                      192 ;
                      193 ;**     LOAD THE LEFT BRANCH FILTER WEIGHTS     **
                      194 ;**     INTO THE FILTER.                        **
                      195 ;
00A0 3A0000     E     196 STEP6:   LDA     NODE
00A3 6F               197          MOV     L,A
00A4 3A0000     E     198          LDA     BRANCH
00A7 67               199          MOV     H,A
00A8 23               200          INX     H
00A9 7E               201          MOV     A,M
00AA 320000     E     202          STA     NOWNOD
                      203 ;
                      204 ;
                      205 ;**     LOAD THE LEFT FACT1 VALUE INTO THE      **
```

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

                     206 ;**      FILTER.                                  **
                     207 ;
  00AD 6F            208          MOV     L,A
  00AE 3A0000    E   209          LDA     VAFT1
  00B1 67            210          MOV     H,A
  00B2 5E            211          MOV     E,M
  00B3 23            212          INX     H
  00B4 56            213          MOV     D,M
  00B5 2B            214          DCX     H
  00B6 EB            215          XCHG
  00B7 220000    E   216          SHLD    FACT1
  00BA EB            217          XCHG
                     218 ;
                     219 ;
                     220 ;**      LOAD THE LEFT FACT2 VALUE INTO THE       **
                     221 ;**      FILTER.                                  **
                     222 ;
  00BB 3A0000    E   223          LDA     VAFT2
  00BE 67            224          MOV     H,A
  00BF 5E            225          MOV     E,M
  00C0 23            226          INX     H
  00C1 56            227          MOV     D,M
  00C2 2B            228          DCX     H
  00C3 EB            229          XCHG
  00C4 220000    E   230          SHLD    FACT2
  00C7 EB            231          XCHG
```

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

                     232 ;
                     233 ;
                     234 ;**      LOAD THE LEFT FACT3 VALUE INTO THE      **
                     235 ;**      FILTER.                                 **
                     236 ;
  00C8 3A0000    E   237          LDA     VAFT3
  00CB 67            238          MOV     H,A
  00CC 5E            239          MOV     E,M
  00CD 23            240          INX     H
  00CE 56            241          MOV     D,M
  00CF 2B            242          DCX     H
  00D0 EB            243          XCHG
  00D1 220000    E   244          SHLD    FACT3
  00D4 EB            245          XCHG
                     246 ;
                     247 ;
                     248 ;**      LOAD THE APPROPRIATE INPUT SCALE        **
                     249 ;**      FACTOR VALUE.                           **
                     250 ;
  00D5 3A0000    E   251          LDA     SCALE
  00D8 67            252          MOV     H,A
  00D9 5E            253          MOV     E,M
  00DA 23            254          INX     H
  00DB 56            255          MOV     D,M
  00DC EB            256          XCHG
  00DD 220000    E   257          SHLD    SCALER
```

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

                    258 ;
                    259 ;
                    260 ;
                    261 ;
                    262 ;**     RESET THE DELAYED INPUTS TO ZERO.       **
                    263 ;
  00E0 210000       264          LXI     H,0
  00E3 220000   E   265          SHLD    DELAY1
  00E6 220000   E   266          SHLD    DELAY2
  00E9 220000   E   267          SHLD    DELAY3
  00EC 220000   E   268          SHLD    DELAY4
                    269 ;
                    270 ;
                    271 ;**     SET THE POINTER TO THE NEXT PORTION OF  **
                    272 ;**     THE ALGORITHM TO BE OPERATED WHEN       **
                    273 ;**     CONTROL IS NEXT RETURNED TO THIS        **
                    274 ;**     ROUTINE.                                **
                    275 ;
  00EF 21FE00   C   276          LXI     H,STEP7
  00F2 220000   E   277          SHLD    NXTSTP
                    278 ;
                    279 ;
                    280 ;**     RESET THE MDS SYSTEM INTTERRUPTS AND    **
                    281 ;**     THE RTI-1200 INTERRUPTS.                **
                    282 ;
  00F5 3E20         283          MVI     A,20H
```

```
LOC  OBJ           SEQ          SOURCE STATEMENT

00F7 D3FD           284          OUT     0FDH
00F9 320000    E    285          STA     STATUS
00FC FB             286          EI
00FD C9             287          RET
                    288 ;
                    289 ;
                    290 ;**      LOAD THE APPROPRIATE THRESHOLD VALUE.    **
                    291 ;
00FE 3A0000    E    292 STEP7:   LDA     NOWNOD
0101 6F             293          MOV     L,A
0102 3A0000    E    294          LDA     THRESH
0105 67             295          MOV     H,A
0106 5E             296          MOV     E,M
0107 23             297          INX     H
0108 56             298          MOV     D,M
                    299 ;
                    300 ;
                    301 ;**      COMPARE THE LEFT OUTPUT PEAK VALUE       **
                    302 ;**      WITH THE THRESHOLD VALUE.                **
                    303 ;
0109 2A0000    E    304 STEP8:   LHLD    PEAK
010C 220000    E    305          SHLD    LEFT
010F 19             306          DAD     D
0110 29             307          DAD     H
                    308 ;
                    309 ;
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

                   310 ;**     RESTORE THE PEAK TO ZERO AND THE COUNT  **
                   311 ;**     24 FOR THE NEXT OUTPUT AVERAGE VALUE    **
                   312 ;**     CALCULATION.                            **
                   313 ;
0111 210000        314         LXI     H,0
0114 220000    E   315         SHLD    PEAK
0117 3E18          316         MVI     A,24D
0119 320000    E   317         STA     COUNT
011C D29E01    C   318         JNC     RECHEK
                   319 ;
                   320 ;
                   321 ;**     COMPARE THE RIGHT AND THE LEFT OUTPUT   **
                   322 ;**     AVERAGE VALUES.  IF THE RIGHT VALUE IS  **
                   323 ;**     GREATER, BRANCH TO THE RIGHT.  IF THE   **
                   324 ;**     LEFT IS GREATER, BRANCH TO THE LEFT.    **
                   325 ;
011F 2A0000    E   326 STEP9:  LHLD    RIGHT
0122 7D            327         MOV     A,L
0123 2F            328         CMA
0124 6F            329         MOV     L,A
0125 7C            330         MOV     A,H
0126 2F            331         CMA
0127 67            332         MOV     H,A
0128 23            333         INX     H
0129 EB            334         XCHG
012A 2A0000    E   335         LHLD    LEFT
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

012D 19           336          DAD     D
012E 29           337          DAD     H
012F D24101    C  338          JNC     STEP11
                  339 ;
                  340 ;
                  341 ;**      SET THE NODE VALUE TO THE RIGHT BRANCH  **
                  342 ;**      VALUE.                                  **
                  343 ;
0132 3A0000    E  344 STEP10: LDA     NODE
0135 6F           345          MOV     L,A
0136 3A0000    E  346          LDA     BRANCH
0139 67           347          MOV     H,A
013A 7E           348          MOV     A,M
013B 320000    E  349          STA     NODE
013E C34E01    C  350          JMP     STEP12
                  351 ;
                  352 ;
                  353 ;**      SET THE NODE VALUE TO THE LEFT BRANCH   **
                  354 ;**      VALUE.                                  **
                  355 ;
0141 3A0000    E  356 STEP11: LDA     NODE
0144 6F           357          MOV     L,A
0145 3A0000    E  358          LDA     BRANCH
0148 67           359          MOV     H,A
0149 23           360          INX     H
014A 7E           361          MOV     A,M
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

014B 320000   E    362           STA     NODE
                   363 ;
                   364 ;
                   365 ;**       CHECK TO SEE IF THE NEW NODE HAS A RIGHT**
                   366 ;**       BRANCH.                                  **
                   367 ;
014E 6F            368 STEP12:   MOV     L,A
014F 3A0000   E    369           LDA     BRANCH
0152 67            370           MOV     H,A
0153 7E            371           MOV     A,M
0154 FEFF          372           CPI     0FFH
0156 C22200   C    373           JNZ     STEP3
                   374 ;
                   375 ;
                   376 ;**       LOAD THE NODE FILTER WEIGHTS INTO THE    **
                   377 ;**       FILTER.                                  **
                   378 ;
0159 3A0000   E    379 STEP13:   LDA     NODE
                   380 ;
                   381 ;
                   382 ;**       LOAD THE NODE FACT1 VALUE INTO THE       **
                   383 ;**       FILTER.                                  **
                   384 ;
015C 6F            385           MOV     L,A
015D 3A0000   F    386           LDA     VAFT1
0160 67            387           MOV     H,A
```

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

  0161 5E            308          MOV     E,M
  0162 23            309          INX     H
  0163 56            310          MOV     D,M
  0164 2B            311          DCX     H
  0165 EB            312          XCHG
  0166 220000    E   313          SHLD    FACT1
  0169 EB            314          XCHG
                     315 ;
                     316 ;
                     317 ;**      LOAD THE NODE FACT2 VALUE INTO THE      **
                     318 ;**      FILTER.                                 **
                     319 ;
  016A 3A0000    E   320          LDA     VAFT2
  016D 67            321          MOV     H,A
  016E 5E            322          MOV     E,M
  016F 23            323          INX     H
  0170 56            324          MOV     D,M
  0171 2B            325          DCX     H
  0172 EB            326          XCHG
  0173 220000    E   327          SHLD    FACT2
  0176 EB            328          XCHG
                     329 ;
                     330 ;
                     331 ;**      LOAD THE NODE FACT3 VALUE INTO THE      **
                     332 ;**      FILTER.                                 **
                     333 ;
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

0177 3A0000   E    414          LDA     VAFT3
017A 67            415          MOV     H,A
017B 5E            416          MOV     E,M
017C 23            417          INX     H
017D 5E            418          MOV     E,M
017E 2B            419          DCX     H
017F EB            420          XCHG
0180 220000   E    421          SHLD    FACT3
0183 EB            422          XCHG
                   423 ;
                   424 ;
                   425 ;**      LOAD THE APPROPRIATE INPUT SCALE FACTOR **
                   426 ;**      VALUE.                                  **
                   427 ;
0184 3A0000   E    428          LDA     SCALE
0187 67            429          MOV     H,A
0188 5E            430          MOV     E,M
0189 23            431          INX     H
018A 56            432          MOV     D,M
018B EB            433          XCHG
018C 220000   E    434          SHLD    SCALER
                   435 ;
                   436 ;
                   437 ;**      RESET THE DELAYED INPUTS TO ZERO.       **
                   438 ;
018F 210000        439          LXI     H,0
```

```
  LOC  OBJ           SEQ          SOURCE STATEMENT

  0192 220000    E    440          SHLD    DELAY1
  0195 220000    E    441          SHLD    DELAY2
  0198 220000    E    442          SHLD    DELAY3
  019B 220000    E    443          SHLD    DELAY4
                      444 ;
                      445 ;
                      446 ;**      SET UP THIS ADAPTIVE ROUTINE TO RECHECK **
                      447 ;**      THE PEAK OUTPUT VALUE OF THE FILTER     **
                      448 ;**      TO VERIFY THAT THE FILTER CENTER        **
                      449 ;**      FREQUENCY IS STILL PROPERLY SET.        **
                      450 ;
  019E 21AD01    C    451 RECHEK: LXI     H,STEP14
  01A1 220000    E    452          SHLD    NXTSTP
                      453 ;
                      454 ;
                      455 ;**      RESET THE MDS SYSTEM INTERRUPTS AND     **
                      456 ;**      THE RTI-1200 INTERRUPTS.                **
                      457 ;
  01A4 3E20           458          MVI     A,20H
  01A6 D3FD           459          OUT     0FDH
  01A8 3A0000    E    460          LDA     STATUS
  01AB FB             461          EI
  01AC C9             462          RET
                      463 ;
                      464 ;
                      465 ;**      SET THE THRESHOLD VALUE FOR COMPARISON  **
```

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

                     466 ;**      WITH THE NEXT OUTPUT PEAK VALUE.        **
                     467 ;
  01AD 210000        468 STEP14: LXI      H,0
  01B0 220000    E   469          SHLD     PEAK
  01B3 3E18          470          MVI      A,24D
  01B5 320000    E   471          STA      COUNT
  01B8 C39E01    C   472          JMP      RECHEK
                     473 ;
                     474 ;
                     475 ;
                     476 ;
                     477 ;
                     478 ;
                     479          END

ASSEMBLY COMPLETE,  NO ERRORS
```